콤퓨터도형학

Computer Graphics

김책공업종합대학출판사
주체91

콤퓨터도형학

김책공업종합대학출판사

차 례

머리말

1장. 콤퓨터도형학개관

2장. 도형처리체계개론

3장. 출력기초요소

4장. 출력기초요소들의 속성

5장. 2차원기하학적변환

6장. 2차원보기

7장. 구조물 및 계층적인 모형화

8장. 도형사용자대면부 및 대화식입력방법

9장. 3차원개념

10장. 3차원물체의 표현

11장. 3차원기하학적변환과 모형화변환

12장. 3차원보기

13장. 보이는 면의 검출방법

14장. 조명모형과 면실감처리

15장. 색모형과 색응용

16장. 콤퓨터동화

부록. 콤퓨터도형처리를 위한 수학

머 리 말

콤퓨터도형학(Computer Graphics)은 사람들의 커다란 관심속에 급속히 발전하는 콤퓨터과학기술분야들중의 하나이다. 이 책의 1판이 나온 이후 콤퓨터도형처리는 사용자대면부, 자료의 가시화, 텔레비죤광고, 영화 및 기타 여러 응용분야들에서 일반적인 구성부분으로 쓰이고 있다. 한편 그림생성의 속도, 현실성, 효과성을 개선하기 위한 하드웨어들과 알고리듬들도 개발되였다. 현재 콤퓨터도형처리에서는 3차원도형처리알고리듬들에 물리학원리들을 더 많이 병합시켜 물체와 조명환경사이의 복잡한 교감을 더 잘 모의하는것이 추세로 되고 있다.

쏘프트웨어규격

국제규격화기구 ISO와 미국규격협회 ANSI에서 첫 도형처리규격으로 GKS(도형처리핵심체계 Graphical Kernel System)를 채택한 이후 도형처리쏘프트웨어의 규격에서는 많은 큰 개선이 있었다. 지금은 PHIGS(프로그람작성자의 계층적 및 대화식도형처리규격 Programmer's Hierarchical Interactive Graphics Standard)가 ISO와 ANSI의 규격으로 되였다. PHIGS와 그 확장판 PHIGS+프로그람들은 광범히 리용되고 있다. 이와 함께 Silicon Graphics GL(Graphics Library), Open GL, Pixar RenderMan대면부, 페지설계를 위한 PostScript해석기 그리고 여러가지 그림그리기, 도면작성, 설계체계들을 비롯한 대중적인 프로그람제품들도 수많이 나왔다.

새로운 내용들

콤퓨터도형학분야에서 커다란 변화들이 일어 났기때문에 2판에서는 1판의 일반적인 체계구성을 유지하면서도 완전히 다시 썼다. 1판에서의 모든 내용들을 현재의 기술적내용들로 확장하였으며 여러가지 새로운 내용들을 많이 보충하였다. 중요하게 확장된 내용들로서는 경계허상제거, 프락탈 및 기타 물체표현법, 광선추적법, 스플라인곡선과 곡면, 조명모형, 면실감처리방법, 콤퓨터동화기술들이다. 2판에서 새롭게 보충된 내용들로서는 가상현실, 도형처리알고리듬의 병렬적실현, 초2차곡면, BSP나무, 형태문법, 립자계, 물리적모형화, 과학적가시화, 업무가시화, 도형처리알고리듬들에서의 4원수법, 분포광선의 추적, 고속퐁명암, 복사세기, 곰보만들기, 화상변화, 도형처리응용에 유용한 여러가지 수학적방법들이 속한다.

이 책은 콤퓨터도형학에 대한 예비지식이 없는 대학생들의 교재로서도 쓸수 있고 도형처리전문가들의 참고서로도 쓸수 있다. 이 책에서는 도형처리체계들을 설계하고 사용하며 리해하는데 필요한 기초원리들을 강조하면서 콤퓨터도형학의 여러가지 응용은 물론이고 도형처리체계의 하드웨어적, 쏘프트웨어적구성요소들을 설명하였으며 C언어로 된 프로그람실례들을 넣어 도형처리알고리듬의 실현과 응용을 보여 주었다. 그리고 알고리듬실현과 도형처리응용을 설명하는 C프로그람에서는 PHIGS와 PHIGS+ 함수들의 사용방법과 PHIGS, PHIGS+, GKS, 기타 도형처리서고들의 특징들도 고찰하고 있다.

요구되는 지식

이 책은 독자들이 아직 콤퓨터도형학에 대하여 알고 있는것은 없지만 콤퓨터프로그람작성과 기본적인 자료구조에 대한 지식은 얼마간 가지고 있다고 가정한다. 콤퓨터도형처리알고리듬들에서는 여러가지 수학적방법들이 리용되는데 이런 방법들을 부록에서 일부 상세하게 해설하고 있다. 부록에 들어 있는 수학적인 내용들은 해석기하, 선형대수, 벡토르 및 텐소르해석, 복소수, 4원수, 수값해석에서 나오는 수법들이다.

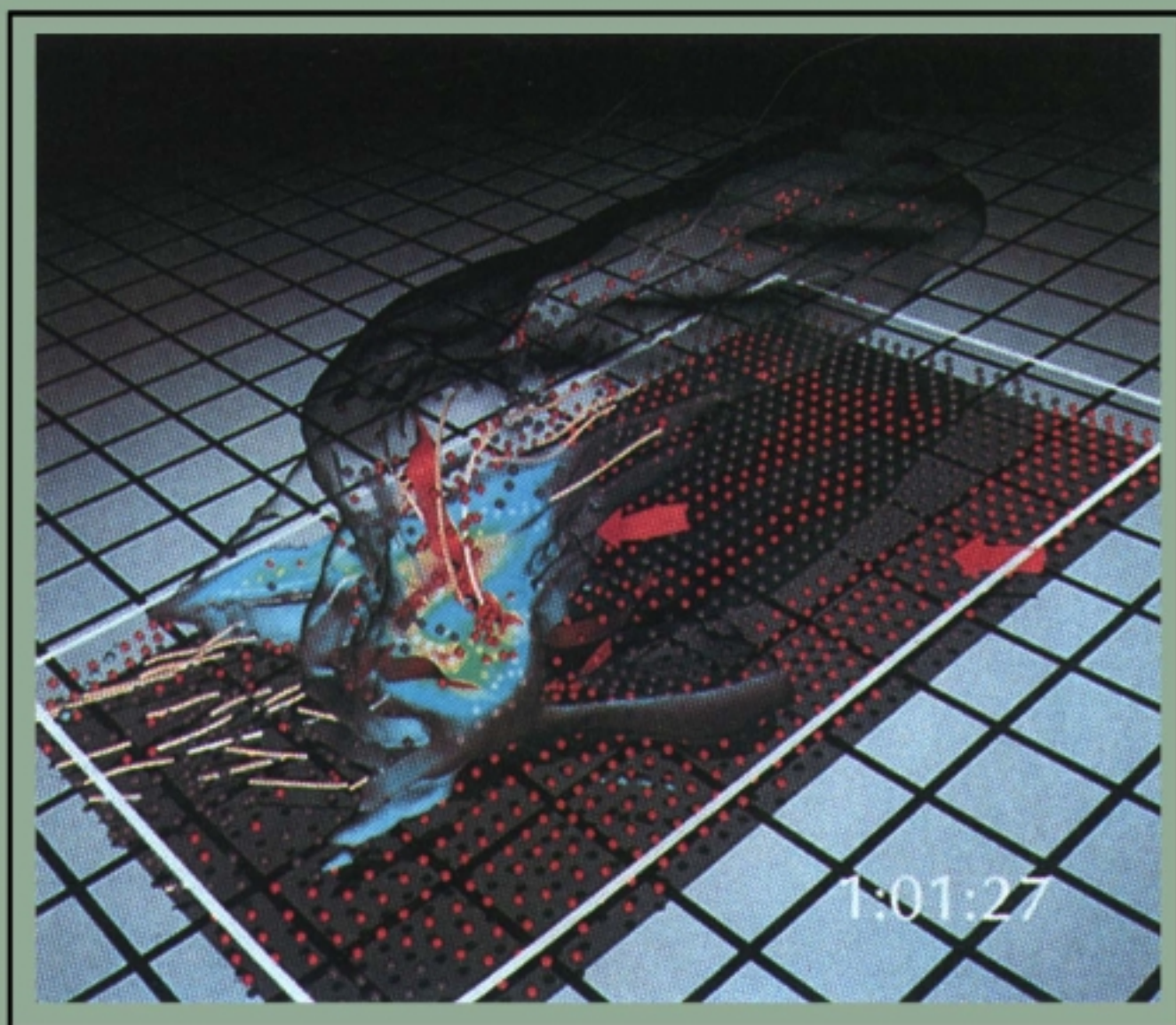
교재로서의 이 책의 리용

2판의 자료들은 필자들이 지난 몇해동안에 걸쳐 콤퓨터도형학입문, 고급도형처리, 과학적가시화, 도형처리연구과정들을 비롯하여 여러 과정들에서 가르친 내용들로부터 전개되였다. 한 학기과정에서는 구체적인 과정의 요구에 따라 2차원적방법들을 취급하거나 혹은 2차원과 3차원의 부분적인 내용들을 결합하여 취급하는 방법을 선택할수 있다. 두 학기과정에서는 첫 학기에 기본적인 도형처리개념들과 방법들을 취급하고 두번째 학기에 고급한 3차원방법들과 알고리듬들을 취급할수 있다. 자습하는 독자들은 처음에 앞장들을 도형처리개념을 가지는데 리용하고 뒤장들에서는 독자들의 흥미에 따라 개별적인 내용들을 선택할수 있다.

대학생준위의 예비적인 2차원도형처리과정에서는 2장부터 8장까지의 기초적인 내용들에 9장에서 주는 3차원적개념들과 내용들을 합쳐 상세하게 취급하도록 조직할수 있다. 색모형, 동화, 스플라인곡선, 2차원프락탈표현 등과 같은 선택적인 내용들은 뒤장들에서 보충적인 자료로 리용할수 있을것이다. 상급학년학생이나 연구생들을 위한 과정에서는 기초적인 2차원개념들과 방법들을 과정의 첫 절반기간에 취급하고 3차원모형화와 보기, 실감처리 등의 선택적인 내용들은 나머지 절반기간에 취급할수 있다. 또 하나의 과정 즉 상급과정에서는 물체의 표현, 면실감처리, 콤퓨터동화내용들을 선택하여 취급할수 있다.

1장은 콤퓨터도형학개관으로서 다양한 응용분야들을 보여 준다. 그다음 2장에서는 도형처리체계의 하드웨어적, 쏘프트웨어적구성요소들에 대하여 소개하며 3장과 4장에서는 2차원도형처리물체들의 표현과 현시에 대한 가장 기초적인 알고리듬들을 준다. 이 두개 장에서는 기본적인 그림요소들을 만드는 방법들과 크기, 색갈 등 물체의 기타 속성들을 조종하는 수법들을 고찰한다. 이것은 학생들에게 도형처리루틴을 실현하는데 필요한 프로그람작성기술들을 안내해 준다. 5장과 6장에서는 2차원기하학적변환과 보기알고리듬들을 취급한다. 7장에서는 2차원그림요소들을 개별적인 구조물로 모형화하고 조직화하는 방법들을 취급하고 8장에서는 사용자대면부의 도형처리적방법들과 가상현실체계를 비롯한 여러가지 도형처리응용에서의 대화식입력방법들을 준다.

9장에서는 3차원기술을 소개하고 10장에서는 3차원물체를 도형처리적으로 표현할수 있는 여러가지 방법들을 물체의 특성에 따라 고찰한다. 11장에서는 3차원적인 모형화와 기하학적변환을 수행하는 방법들을 주며 12장에서는 3차원장면의 보임상을 얻기 위한 방법들을 자세히 언급한다. 장면안에서 보이는 면식별에 대한 여러가지 알고리듬들은 13장에서 고찰하며 광선추적, 복사세기와 같은 조명모형화, 면실감처리방법들은 14장에서 취급한다. 색모형과 방법들은 15장에서, 동화기술은 16장에서 고찰한다.

1장. 콤퓨터도형학개관

콤퓨터는 그림을 빠르게 그리고 효과적으로 만들어 내는 강력한 도구로 리용되고 있다. 도형적인 화면들을 효과적으로 쓸수 없는 분야란 사실상 없다. 때문에 콤퓨터도형학이 그처럼 광범하게 리용되고 있는데 대하여 놀라워 할것은 조금도 없다. 초기 과학과 기술에서의 응용은 값이 비싸고 복잡한 설비들에 의거해야 했지만 콤퓨터기술이 발전함에 따라 대화식콤퓨터도형처리가 실천적인 도구로 되게 되였다. 오늘 콤퓨터도형처리는 과학, 기술, 의학, 사무, 공업, 정치, 예술, 오락, 광고, 교육, 훈련과 같은 여러 분야들에서 일상적으로 리용되고 있다. 그림 1-1은 모의, 교육, 그라프표시에서의 몇가지 도형처리응용을 보여 주고 있다. 콤퓨터도형처리가 어떻게 진행되는가 하는 세부에 들어 가기전에 도형처리응용전람관을 간단히 참관하기로 한다.

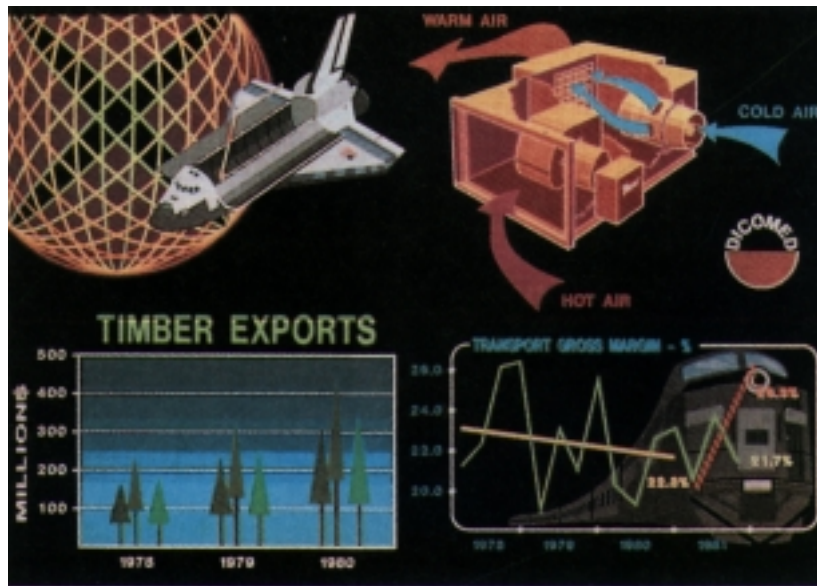



그림 1-1. 콤퓨터도형처리의 응용실례

1절. 콤퓨터지원설계

콤퓨터도형학은 실천적으로 공학 및 건축체계에 대한 설계공정에서 많이 리용되고 있다. 지금은 거의 모든 제품들이 콤퓨터로 설계되고 있는데 건물, 자동차, 비행기, 함선, 우주비행선, 콤퓨터, 직물 그리고 기타 수많은 제품들의 설계에서는 일반적으로 CAD(Computer–Aided Design)라고 하는 콤퓨터지원설계방법들이 일상적으로 리용되고 있다.

일부 설계분야에서는 먼저 물체들을 그의 전체적인 형태와 내부특징을 보여 주는 선그물구조의 륜곽선형식으로 현시한다. 선그물구조화면은 설계자들이 설계형태를 대화적으로 조정할 때 그 효과들을 빨리 볼수 있게 한다. 그림 1-2와 1-3에는 설계에서 선그물구조화면의 응용실례를 주었다.

CAD응용을 위한 쏘프트웨어패키지들은 그림 1-4와 1-5에서와 같이 일반적으로 설계자에게 다중창문환경을 제공한다. 현시된 여러가지 창문들은 물체의 확대된 상이나 기타 다른 상을 보여 준다.

그림 1-5에 보여 준바와 같이 회로들과 통신망, 물공급망, 기타 편의망들은 몇가지 도형적인 형태들의 반복배치에 의하여 만들어 진다. 설계에 리용되는 형태들은 망 또는 회로안의 각이한 요소들을 표현한다. 전기, 전자, 론리회로들에 대한 표준적인 형태들은 흔히 설계프로그람들이 준다. 설계자는

다른 응용을 위하여 망 또는 회로를 만드는데 리용될 개별적인 부호들을 새로 만들수도 있다. 그다음 도면에 요소들을 잘 배치하여 체계를 설계하는데 도형처리프로그람들은 요소들사이의 련결을 자동적으로 진행한다. 이것은 설계자가 회로들을 빨리 교체해 보면서 요소들의 개수나 체계가 요구하는 공간을 최소로 할수 있게 한다.

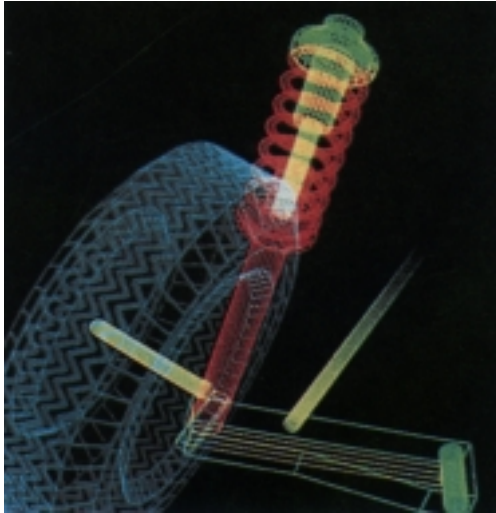



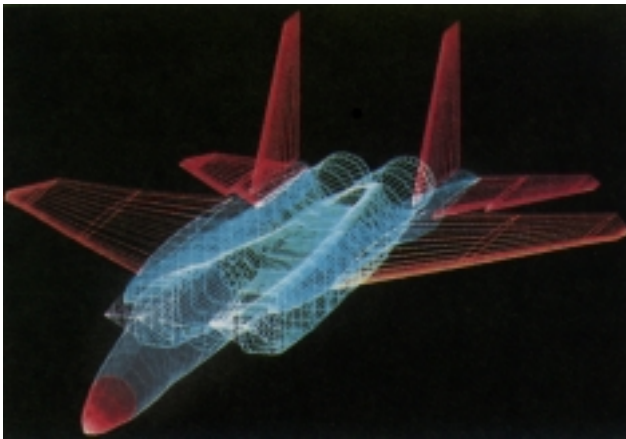
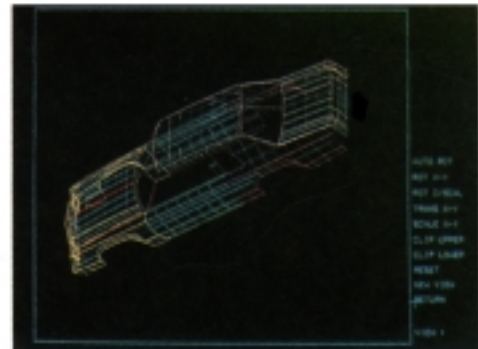
그림 1-2. 자동차바퀴조립품에 대한 색부호화된 선그물구조화면

ㄱ)

ㄴ)

그림 1-3. 비행기와 자동차의 본체설계의 색부호화된 선그물구조화면

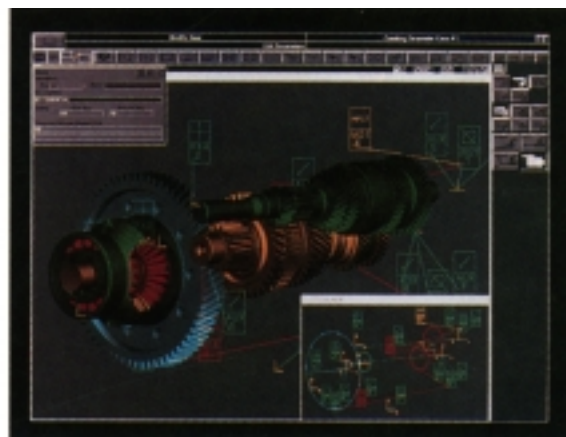
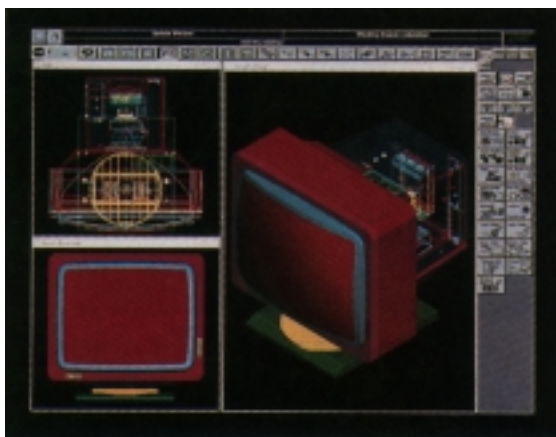



그림 1-4. 다중창문(색부호화된 CAD 워크스테이션화면)

그림 1-5. 회로설계에서의 응용(고성기와 마이크가 접속된 SUN 워크스테이션에서 다중창문과 색부호화된 론리요소를 리용하고 있다.)

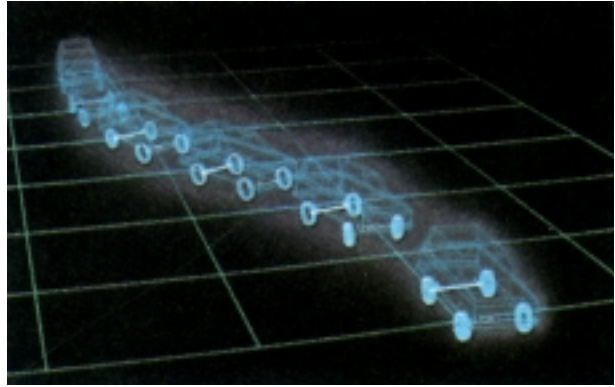



그림 1-6. 길이 변할 때 운수수단의 움직임에 대한 모의

동화는 CAD분야에서 자주 쓰인다. 영상현시장치에서 선그물구조화면을 리용하는 실시간동화는 그림 1-6에서 보여 주는바와 같이 운수수단이나 체계의 움직임을 검사하는데 쓸모 있다. 물체현시에서 실감처리된 면을 쓰지 않으면 동화의 매개 토막에 대한 계산량이 적기때문에 화면에서 원활한 실시간운동을 얻을수 있다. 또한 선그물구조화면은 설계자가 운수수단의 내부를 들여다 보며 운동시 내부요소들의 움직임을 볼수 있게 한다.

가상현실환경에서의 동화는 어떤 운동이 운수수단의 조작자에게 어떻게 영향을 미치는가를 결정하는데 리용된다. 그림 1-7에서 모자는 뜨락또르운전수가 운전을 진행할 때 마치 운전수가 뜨락또르좌석에 앉아서 보는것처럼 앞에 있는 적재바가지나 뒤에 있는 보습의 립체상(그림1-8)을 보여 준다. 이것은 설계자가 운전수의 보기를 방해할수 있는 바가지나 보습의 여러가지 위치를 조사할수 있게 한다. 그러면 전반적인 뜨락또르설계에 고려될수 있다. 그림 1-9는 실제적인 3차원장면대신에 표준영상현시장치에 현시되는 뜨락또르좌석으로부터 합성된 넓은 시각상을 보여 주고 있다. 그리고 그림 1-10에는 개별적인 창문이나 다른 현시장치에 현시될수 있는 뜨락또르의 보임상을 보여 주었다.

그림 1-7. 가상현실환경에서 뜨락또르의 운전(운전이 진행되는 동안 운전수는 모자를 통하여 앞적재기, 뒤보습과 주위를 볼수 있다.)

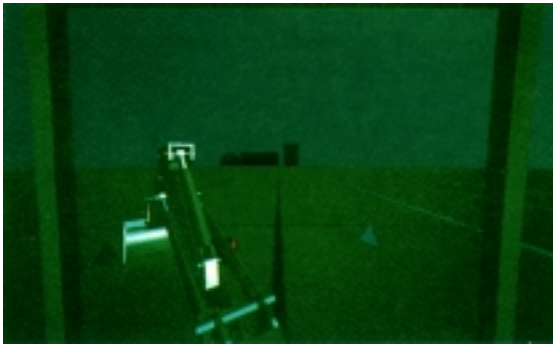



그림 1-8. 뜨락또르운전수가 모자를 통하여 보는 뒤보습

그림 1-9. 운전수가 보는 뜨락또르바가지(표준현시장치에 넓은 시각상을 형성하기 위하여 여러가지 부분들을 합성하였다.)

그림 1-10. 표준현시장치에 현시된 뜨락또르보임상

물체의 설계가 완성되였거나 또는 거의 완성되였을 때 최종제품의 모양을 보여주는 화면을 만들 때에는 현실감을 부여하는 조명모형과 면실감처리기술을 적용한다. 그림 1-11에 이 실례를 보여 주었다. 현실감이 있는 화면은 또한 특수조명효과와 배경장면을 리용하여 자동차와 다른 운수수단들을 광고하기 위하여서도 만든다(그림 1-12).

ㄱ)

ㄴ)

그림 1-11. 현실감 있게 실감처리된 설계제품

제품제작을 자동화하는데서는 제작공정을 설계되는 물체의 콤퓨터서술과 결합시킨다. 실례로 회로기판설계도는 그것을 제작하는데 필요한 개별적인 공정들의 서술로 변환되게 된다. 일부 기계부분품들은 면을 공구로 어떻게 형성하겠는가 하는 서술에 의하여 제작된다. 그림 1-13에서는 물체가 제작될 때 걸면우에서 공구가 취하는 경로를 보여 준다. 그러면 수자조종공작기계의 공구가 이 제작도

면에 따라 부분품을 제작하도록 설정된다.

건축가들은 방, 문, 창문, 계단, 당반, 매대의 배치와 기타 건물면모를 보여 주는 그림 1-14와 같은 층설계도를 그려 내는데 대화식도형처리방법들을 사용한다. 전기기사는 영상현시장치의 건물설계화면우에서 배선을 진행하고 전기접속구, 화재경보체계를 배치해 볼수 있다. 또한 사무실내 또는 작업장의 공간리용을 결정하는 설계에는 편리한 설계프로그람들을 적용할수 있다.

그림 1-12. 완성품에 대한 광고그림(작업장조명효과와 현실감을 부여하는 면실감처리기술을 적용하였다.)

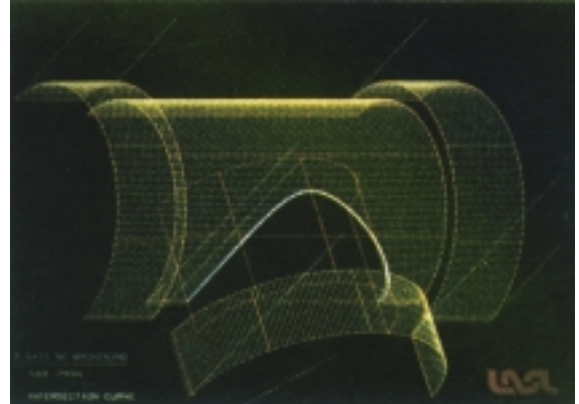



그림 1-13. 부분품의 수자조종기계가공서술에 대한 CAD 도면(부분품의 면은 한색으로 현시되고 공구의 면은 다른 색으로 현시된다.)

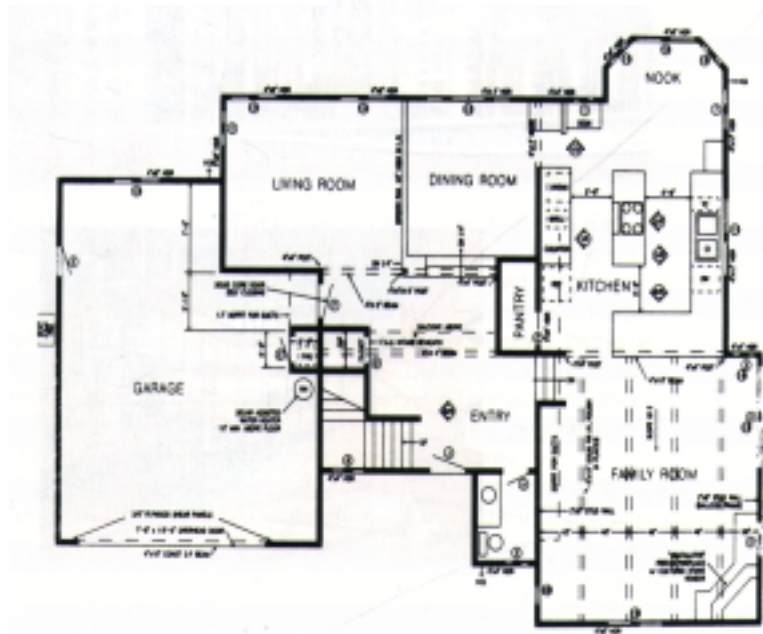



그림 1-14. 건물설계를 위한 건축 CAD 도면

그림 1-15와 같은 건축설계의 현실감 있는 화면들에 의하여 건축가와 그 주문자는 함께 집정원이나 공업지구와 같은 하나 또는 여러 건물들의 집합된 모양을 보고 연구할수 있다. 가상현실세계에서는 설계자들이 지어 방안이나 건물주변을《거닐면서》 설계의 전체 효과를 구체적으로 더 잘 알아 볼수도 있다. 건축 CAD프로그람들은 또한 현실적인 건물들의 외부화면뿐아니라 3차원적인 내부설계와 조명실험도 편리하게 해준다(그림 1-16).

기타 많은 종류의 체계와 제품들이 일반 CAD프로그람들이나 특별히 개발된 CAD쏘프트웨어를 리용하여 설계된다. 실례로 그림 1-17에서는 CAD체계로 설계한 주단무늬를 보여 주었다.

ㄱ)

ㄴ)

그림 1-15. 건물설계의 현실적인 3 차원실감처리
ㄱ-거리에서 본 설계된 세계무역쎈터의 전경, ㄴ-콤퓨터동화를 위해 만든 중앙홀의 구상

그림 1-16. 물결형경로를 따라 벽전등들을 설치하여 움직이는 느낌을 주고 매개 호텔방앞에 현등을 켜서 입구감을 주는 호텔복도

그림 1-17. 콤퓨터도형설계방법으로 만든 동양주단무늬

2 절. 연시도형처리

콤퓨터도형학은 연구조사보고의 삽화를 만들거나 35mm환등필림, 투영필림을 만드는 연시도형처리(Presentation graphics)에 리용되고 있다. 연시도형처리는 연구조사보고, 관리보고, 소비자정보불레찐, 기타 다른 형의 보고들에 대한 재정적, 통계적, 수학적, 과학적, 경제적자료들을 개괄하는데 공통적

으로 리용된다. 화면을 35mm환등필림이나 투영필림으로 변환하기 위한 워크스테이션장치들과 봉사단위들도 있다. 연시도형처리의 대표적인 실례로는 기둥도표, 선그라프, 면그라프, 원도표 그리고 여러 파라메터들사이의 관계를 보여 주는 기타 다른 화면들을 들수 있다.

그림 1-18에서는 지리학적정보가 결합된 2차원도형처리의 실례를 보여 주었다. 이 그림은 하나의 그라프, 세 부분으로 된 원도표와 결합된 세가지 색갈로 부호화된 기둥도표를 보여 주었다. 이런 그라프와 도표는 3차원적으로 현시하면 보충적인 정보를 줄수 있다. 3차원그라프는 때때로 대단히 효과적이다. 그것들은 자료관계를 보다 극적이며 보다 매혹적으로 보여 줄수 있다. 그림 1-19의 도표는 3차원기둥도표와 원도표를 보여 주었다.

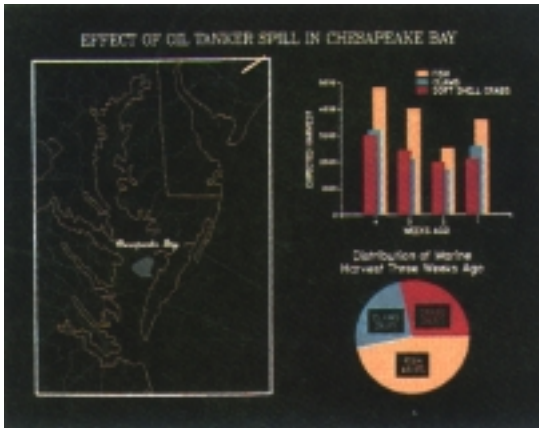



그림 1-18. 지리학적도표와 결합된 2차원막대그라프와 원도표

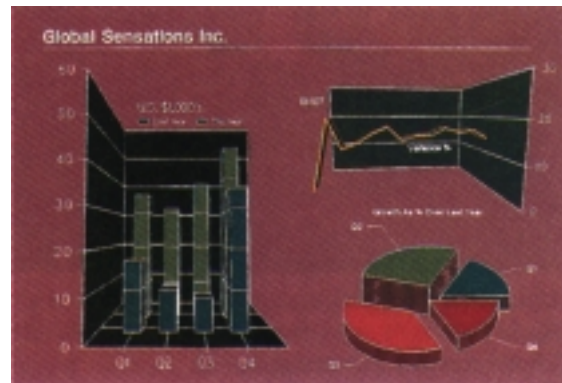



그림 1-19. 3차원기둥도표, 원도표 및 선그라프

3차원그라프의 보충적인 실례를 그림 1-20과 1-21에 보여 주었다. 그림 1-20은 한가지 종류의 걸면도를 보여 주고 그림 1-21은 립체적인 높이와 함께 표시되는 2차원등고선도를 보여 주었다.

그림 1-22에는 과제집행계획에 사용되는 시간도표를 보여 주었다. 시간도표와 과제설계도는 과제의 집행을 계획하고 감시하기 위한 연구과제관리에 리용된다.

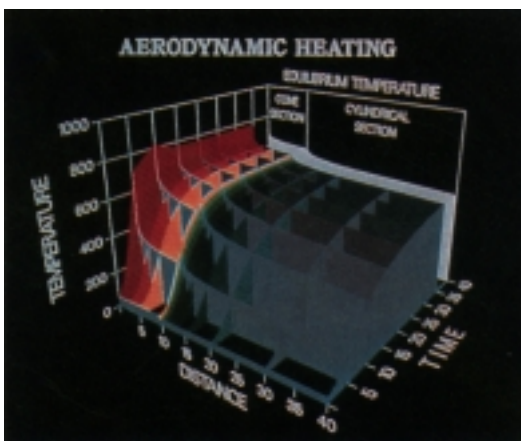



그림 1-20. 걸면도표에 의한 관계 보여 주기

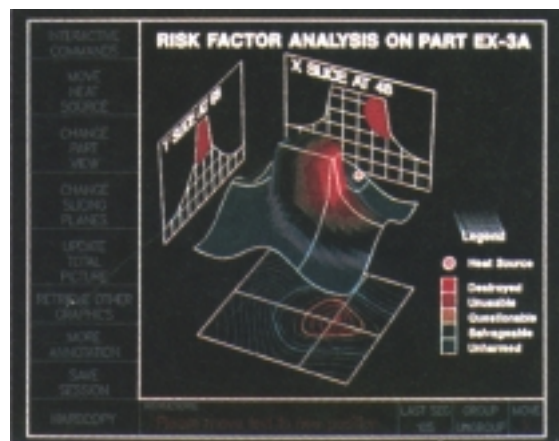



그림 1-21. 지면의 립체적인 높이와 함께 그린 2차원등고선

3 절. 콤퓨터미술

콤퓨터도형학적방법들은 미술작품과 광고미술분야에서도 널리 리용되고 있다. 미술가들은 전용하드웨어, 미술가용그림그리기프로그람(례:Lumena)과 기타 그림그리기프로그람(례:PixelPaint와 Super Paint)들, 특별히 개발된 쏘프트웨어, 기호적인 수학프로그람(례:Mathe-matica), CAD프로그람, 탁상출판쏘프트웨어, 물체형태의 설계와 물체운동을 쉽게 지적할수 있게 하는 동화프로그람들을 비롯하여 여러가지 콤퓨터도형처리방법들을 리용한다.

그림 1-23은 미술가들이 영상현시장치의 화면우에서 그림을 그릴수 있는것은 그림그리기프로그람이 리면에 숨어 있다는 근본적인 내용을 보여 준다. 실제로 그림은 보통 펜을 리용하여 도형입력판(수자화기)우에서 전자적으로 그려 지는데 여러가지 붓놀림, 붓너비, 색갈을 모의할수 있다. 그림 1-24는 그림그리기프로그람을 써서 그린것인데 그림안의 인물은 그림을 그리는데 열중하고 있다.

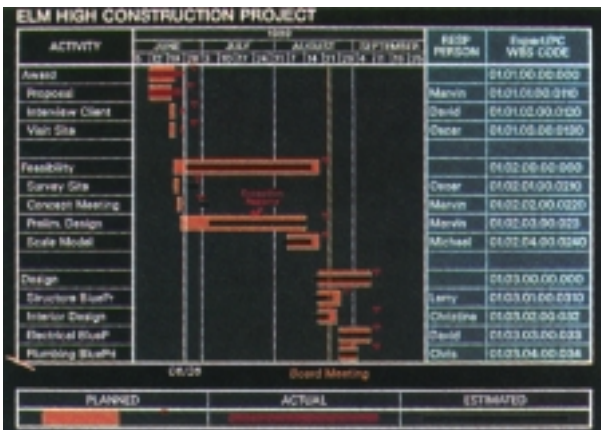



그림 1-22. 연구과제에 대한 관계정보를 현시하는 시간도표

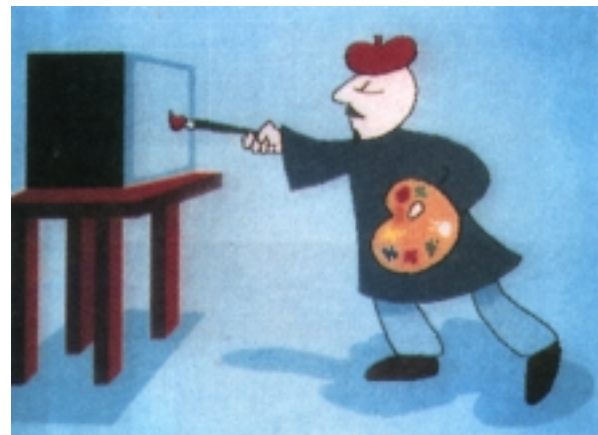



그림 1-23. 영상현시장치에서 작업하는 미술가를 상징적으로 보여 주는 그림그리기프로그람으로 만든 만화

ㄱ)

ㄴ)

그림 1-24. 미술가가 그림그리기체계로 그림을 그리는 만화표현

ㄱ-도형입력판우에서 그린 그림을 영상현시장치에서 꼬마요정들이 구경하는것같이 현시되여 있다, ㄴ-비데오카메라로 체계에 입력하여 적당한 크기로 만들어 놓은 Saint Nicholas 의 유명한 《Thomas Nast》그림우에 이 만화를 덧놓았다.

Van Gogh의 붓놀림을 모의한 그림 1-25의 전자그림을 만드는데는 코드 없는 압력수감펜을 가진 그림그리기체계가 리용되였다. 펜은 손의 압력변화를 선너비, 붓의 크기, 색의 점차적인 변화로 바꾸어 준다. 그림 1-26에서는 이 펜과 수채화구, 파스텔, 유화붓효과를 얻게 하는 쏘프트웨어로 만든 수채화그림을 보여 주었다. 그림에는 여러가지 수채화기법이 잘 모의되였다.

그림 1-27은 화상스캐너로 입력한 화상과 결합된 그림그리기방법들의 실례를 주었다.

그림 1-25. 도형처리미술가 Elizabeth O'Rourke 가 코드 없는 압력수감펜을 가지고 만든 Van Gogh 의 그림

그림 1-26. 코드 없는 압력수감펜과 Lumena gouachebrush 쏘프트웨어를 리용하여 그린 전자수채화

그림 1-27. 전자사태라고 부르는 그림(미술가는 도형입력판과 Lumena 쏘프트웨어를 리용하여 현실세계의 복잡성을 표현하고 있다. 잎사귀,꽃잎, 전자요소들의 그림을 화상스캐너로 입력한 화상과 결합시켰다.)

섬세한 미술가는 화상을 만들기 위하여 여러가지 다른 콤퓨터기술들도 리용한다. 그림 1-28에 보여 준것과 같은 그림을 만들기 위하여 미술가는 3차원모형화프로그람, 결문양입히기 및 작도프로그람, CAD 쏘프트웨어를 결합하여 쓰고 있다. 그림 1-29 는 미술가없이 자동미술을 창조할수 있게 특별히 설계된 쏘프트웨어를 리용하여 펜작도기로 그린 그림이다.

그림 1-28. Whigmalaree라는 전자그림(이 그림은 Spheres of Influence 체계에서 도형입력판과 3차원 모형화, 걸문양입히기,기타 일련의 변환들을 결합하여 쓰는 방법으로 만들었다.)

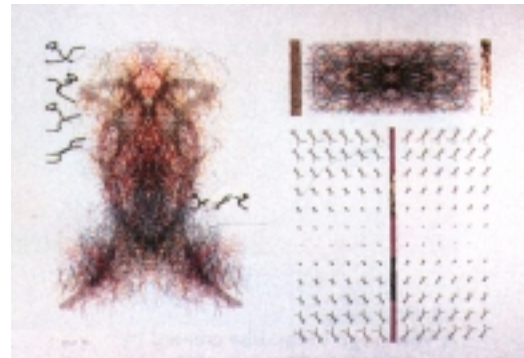



그림 1-29. 미술가가 자기 형식을 모방하도록 특별히 설계한 쏘프트웨어로써 펜작도기에 출력한 전자미술작품(펜작도기는 여러가지 펜과 붓을 비롯한 그리기도구를 가지고 있다.)

그림 1-30에서는 수학미술의 실례를 보여 주었다. 미술가는 수학함수, 프락탈절차, Mathematica 쏘프트웨어, 잉크분사식인쇄기 기타 체계들을 결합하여 여러가지 2차원 및 3차원형태들과 립체화상쌍을 만들고 있다. 수학관계식들을 리용하여 만든 전자미술의 다른 실례를 그림 1-31에 보여 주었다. 이 사람은 자주 시각적 및 청각적무늬를 합성한 영상을 만들고 있는데 그의 미술작품은 음악합성에서의 주파수변화와 기타 파라메터들과의 관계속에서 설계되고 있다.

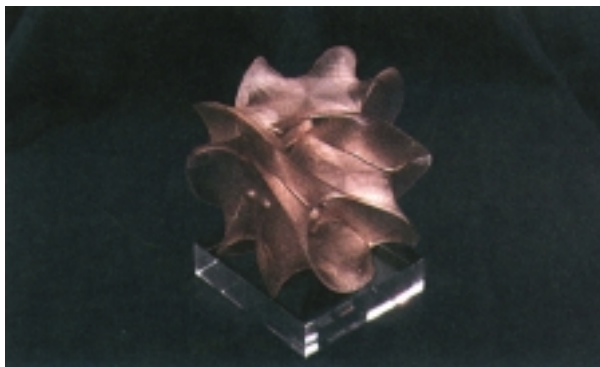



그림 1-30. 이 작품은 Fermat의 Last정리 $x^n + y^n = z^n$, (n=5)을 가시화한데 기초하여 만들었다. 화상은 Mathematica 및 Wavefront 쏘프트웨어로 실감처리하였다.

그림 1-31. 이 미술가-작곡가는 수학함수, 프락탈절차, 초고속콤퓨터를 리용한 음악합성으로 형태와 색갈을 혼합하는 여러가지 설계방안들을 실험하고 있다.

지금까지는 미술작품들에서 전자적인 화상을 만드는 현재의 기술들을 보았다. 이러한 방법들은 또한 마크도안과 기타 설계들에 대한 광고미술, 본문과 도형이 결합된 페지설계, TV광고물 기타 다른 분야들에도 적용된다. 본문과 도형이 결합된 페지설계를 진행하는 워크스테이션을 그림 1-32에서 보여 주었다.

대부분의 광고미술분야(광고영화을 비롯하여)에서는 제품의 화상에 사진 같은 감을 내게 하는 실감처리기술이 리용된다. 그림 1-33에서는 마크도안설계의 실례를, 그리고 그림 1-34에서는 제품광고를 위한 3개의 콤퓨터도형처리화상을 주었다. 또한 동화가 광고에 자주 리용된다. 텔레비죤광고는 프레임별로 만들어 진다. 여기서 운동의 매개 프레임은 실감처리된 다음 화상파일로 보관된다. 매개 련속

된 프레임에서 운동은 물체의 위치를 이전 프레임에서의 위치로부터 약간 움직여 놓는것으로 모의한다. 동화흐름안의 모든 프레임들이 실감처리되면 프레임들은 필림에 옮겨 지거나 재생용영상완충기에 기억된다. 영화필림은 동화흐름에서 초당 24프레임이 보장되여야 하며 동화를 영상현시장치에서 돌린다면 초당 30프레임이 보장되여야 한다.

광고에 많이 쓰이는 공통적인 도형처리방법은 련속형태변화(Morphing)법이다. 여기서는 한 물체가 다른 물체로 변화(변형)된다. 이 방법은 TV광고에서 기름통을 자동차기관으로, 자동차를 범으로, 물웅뎅이를 머리장식으로, 한 사람의 얼굴을 다른 얼굴로 변화시키는데 리용된다. 련속형태변화의 실례를 그림 1-40에 보여 주었다.

그림 1-32. 페지설계워크스테이션

그림 1-33. 3차원실감처리된 상표마크

ㄱ)

ㄴ)

ㄷ)

그림 1-34. 제품광고

4절. 오락

지금 영화, 음악비데오, 텔레비존련속물들을 만드는데서 콤퓨터도형학적방법들이 공통적으로 리용되고 있다. 도형처리한 장면들만 독자적으로 현시되기도 하고 도형처리한 물체들이 배우나 실지장면들과 합성되기도 한다.

영화 《Star Trek－The Wrath of Khan》를 위하여 만든 도형처리장면을 그림 1-35에 보여 주었다. 행성과 우주비행선은 선그물구조형식으로 그려 져 있는데 후에 립체면을 만드는 실감처리방법으로 명암을 주게 된다.

그림 1-36에 고급한 모형화와 면실감처리을 가지고 만든 평가 받은 2개의 단편영화의 장면들을 보여 주었다.

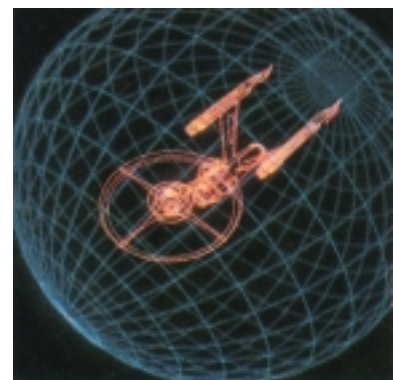



그림 1-35. Paramount Picture 영화 《Star Trek － The Wrath of Khan》를 위하여 개발한 도형처리화면

많은 TV 련속물들에서 콤퓨터도형학적방법들을 정상적으로 리용하고 있다. 그림 1-37에서는 련속물 《Deep Space Nine》을 위하여 만든 장면을 보여 주었다. 그리고 그림 1-38은 련속물《Stay Tuned》에서 실제장면의 배우들과 합성된 선그물구조의 사람을 보여 준다. 그림 1-39는 일본 TV방영을 위하여 만든 13세기 Dadu(지금의 베이징)의 복원화상인데 대단히 실감 있는 화상이다.

ㄱ)

ㄴ)

그림 1-36. ㄱ-영화 《Red's Dream》에서 나오는 콤퓨터로 만든 장면, ㄴ-영화《Knickknack》에서 나오는 콤퓨터로 만든 장면

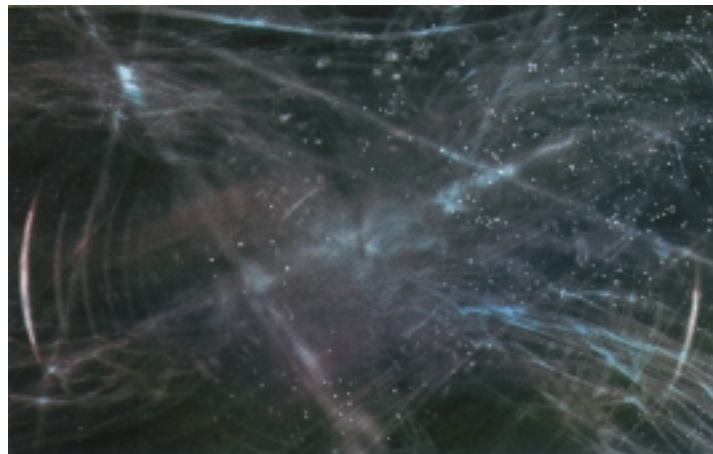



그림 1-37. TV 련속물 《Deep Space Nine》에서 도형처리한 장면

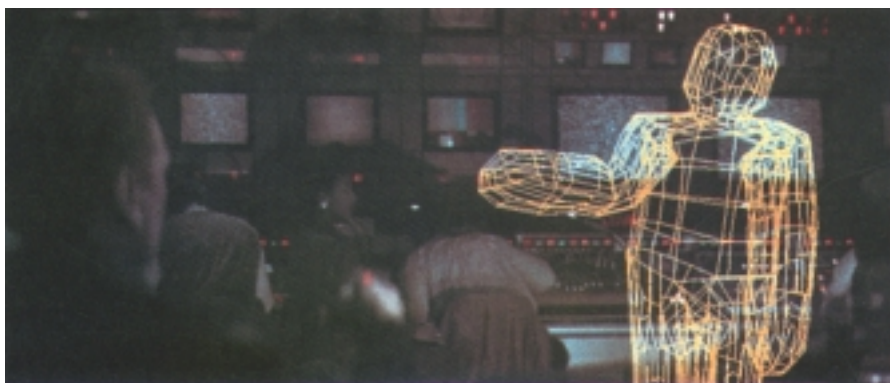



그림 1-38. TV 련속물 《Stay Tuned》의 실지장면과 결합된 도형처리기술

그림 1-39. 13세기의 Dadu(오늘 베이징)의 복원화상

음악비데오에서는 도형처리를 여러가지 방법으로 리용한다. 그림 1-38에서와 같이 도형처리물체들이 실지동작과 결합될수도 있고 도형처리와 화상처리기술을 리용하여 사람이나 물체를 다른것으로 변화시킬수도 있다. 련속형태변화의 실례를 그림 1-40 에서 련속장면으로 보여 주고 있다. 이것은 비데오 《She's Mad》에 대하여 만들어 진것이다.

그림 1-40. 비데오《She's Mad》에서의 련속형태변화실례

5절. 교육 및 훈련

콤퓨터가 만들어 내는 물리적, 재정적, 경제적체계들의 모형들은 흔히 교육을 위한 방조수단으로 리용된다. 그림 1-41 의 색부호화된 계통도와 같은 물리학체계, 생리학체계, 인구추세 또는 설비의 모형들은 훈련생들이 체계의 동작을 리해하는것을 도와 주게 된다.

훈련을 목적으로 하는 전문적인 체계들이 설계되고 있다. 이런 전문화된 체계들의 실례로서는 실기수업이나 또는 선장, 비행기조종사, 중설비조작자, 항공운수조종인원들의 훈련을 위한 모의기들을 들수 있다. 일부 모의기들은 영상장면이 없다. 실례로 비행설비의 조종판만 가지고 있는 비행모의기를 들수 있다. 그러나 대부분의 모의기들은 시각적인 조작을 위한 도형처리화면들을 제공하고 있다. 내부보기체계를 가지고 있는 대규모모의기의 두가지 실례를 그림 1-42 와 1-43 에 보여 주었다.

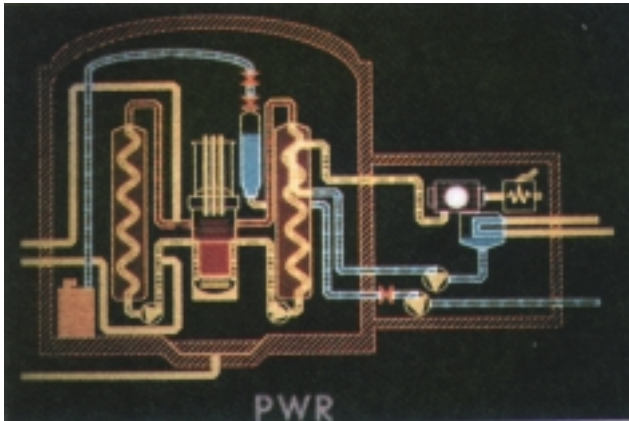



그림 1-41. 원자로의 동작을 설명하는데 리용되는 색부호화된 계통도

그림 1-42. 완전색시각체계와 운동시 6 개 자유도를 가지는 대규모비행모의기

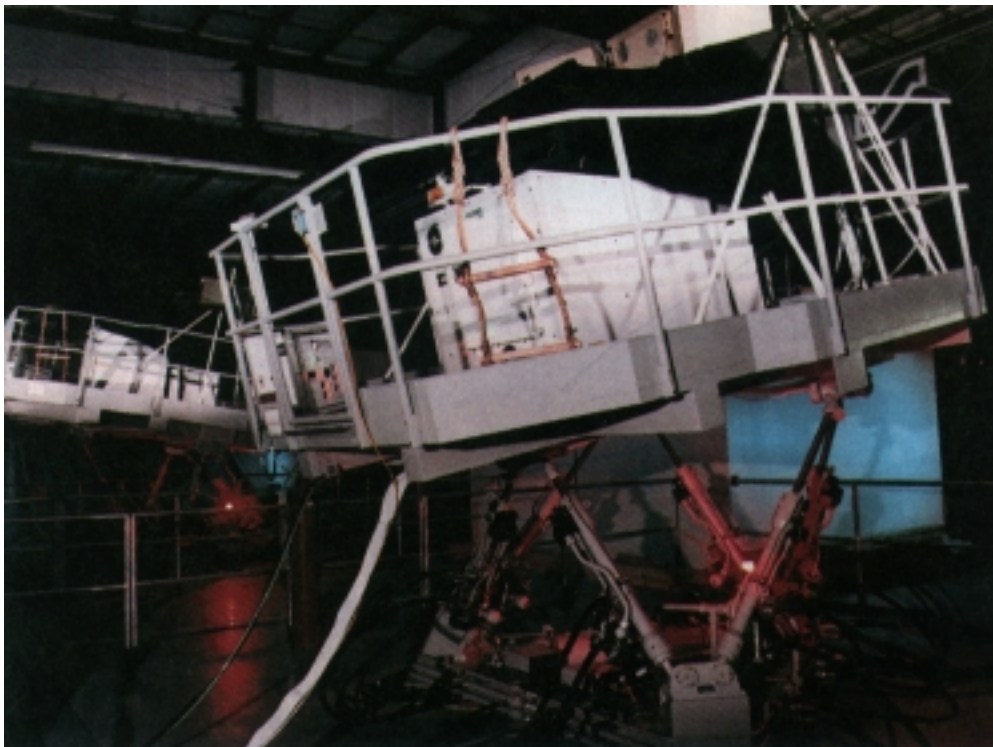



그림 1-43. 시각적인 화상체계를 가지고 있는 군용땅크모의기

보기체계의 다른 형태를 그림 1-44 에 보여 주었는데 여기서 여러가지 계기판이 있는 보기화면은 모의기앞에 설치되고 색투영기들은 영사막에 비행장면을 현시한다. 이런 보기체계가 비행기조종탑성원들의 훈련을 위한 모의기에서도 리용되고 있다. 그림 1-45 는 비행모의기에서 교원이 있는 구역의 실례를 보여 주고 있다. 건반은 비행기동작 또는 환경에 영향을 미치는 파라메터들을 입력하는데 리용되며 펜작도기는 훈련시 비행기의 비행경로를 표시하는데 리용된다.

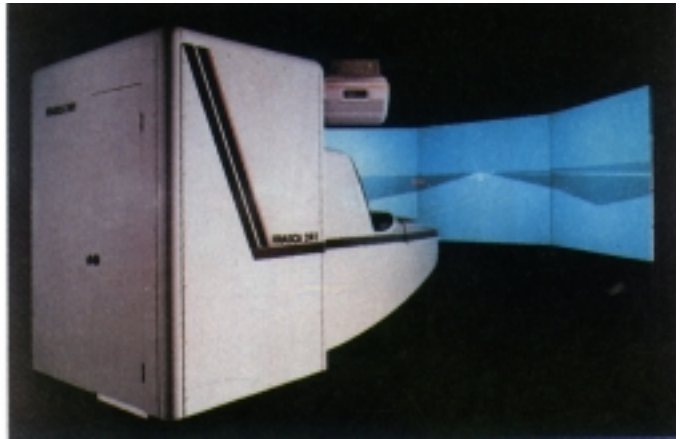



그림 1-44. 외부에 완전색보기체계를 가지고 있는 비행모의기

그림 1-45. 비행모의기안에서 교원의 구역(설비들은 교원이 비행상태를 감시하고 비행기와 환경파라메터들을 설정할수 있게 한다.)

여러가지 모의기들에서 만들어 지는 장면들을 그림 1-46부터 그림 1-48에 보여 주었다. 자동차운전모의기의 출력장치를 그림 1-49에 주었다. 이 모의기는 위험한 환경에서 운전수의 행동을 조사하는데 리용된다. 운전수의 반응은 교통안전을 최대로 보장하는 운수수단의 최적화설계의 기초로 리용된다.

그림 1-46. 비행모의기에서 볼 때의 한 장면

그림 1-47. 군함모의기안에서 볼 때의 한 장면

그림 1-48. 우주왕복비행선안에서 볼 때의 한 장면

그림 1-49. 운전수의 반응을 조사하는데 리용되는 자동차모의기안에서 보는 한 장면

6절. 가시화

과학자, 기사, 의료일군, 업무분석가를 비롯하여 사람들은 흔히 많은 량의 정보를 분석하거나 일정한 과정의 움직임을 연구하게 된다. 초고속콤퓨터에서 진행되는 수값모의에서는 흔히 수천 지어 수백만개의 자료값들을 포함하는 자료파일이 만들어 진다. 이와 류사하게 위성카메라와 기타 정보원천들에서는 미처 해석할수 없을 정도로 대규모자료파일들을 보낸다. 이런 큰 수값모임을 주사하여 경향성과 호상관계를 결정한다는것은 지루하고도 비효과적인 과정이다. 그런데 만약 이런 자료를 시각적인 형식으로 변환하면 흔히 그 경향성과 모양이 명백해 진다. 그림 1-50에서는 대규모자료모임을 지면우에서 상대높이의 색부호화된 화면으로 변환한 실례를 보여 주었다. 일단 이런 방법으로 자료값을 그려 놓으면 자료의 전반적인 모양을 쉽게 알아 볼수 있다. 과학과 공학, 의학자료모임들에 대하여 도형적인 표현을 만들고 처리하는 과정을 일반적으로 **과학적가시화**(Scientific Visualiztion)라고 한다. 그리고 상업, 기업 기타 자연과학밖의 다른 분야들에서 제기되는 자료모임과 관련하여서는 **업무가시화**(business Visualization)라는 용어를 사용한다.

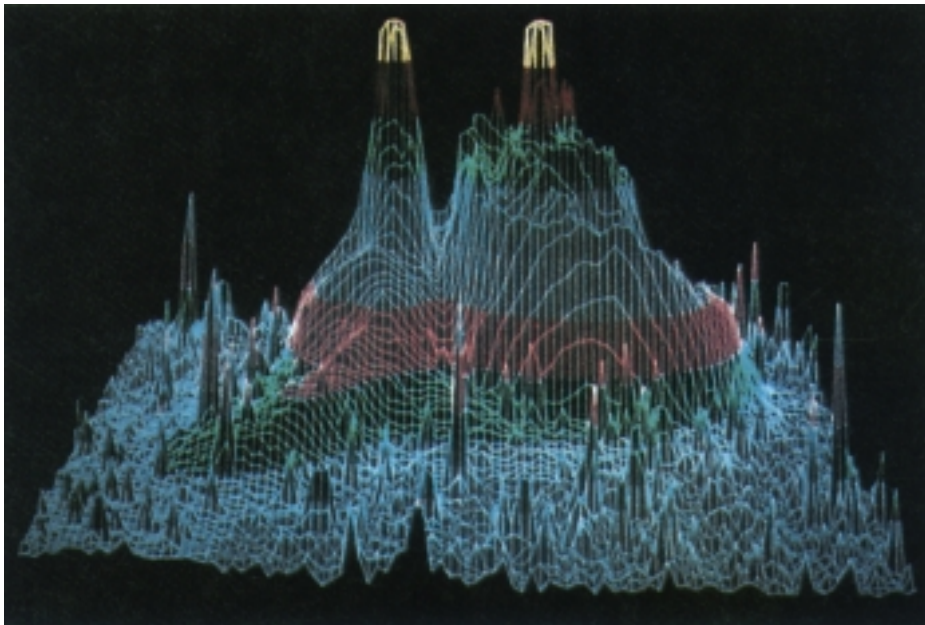



그림 1-50. 소용돌이별구름(Whirlpool Nebula)에 대하여 관찰한 1600 만개 점들의 상대밝기를 색부호화하여 표현한 그림 (두개의 서로 다른 은하수를 보여 주고 있다.)

자료모임의 형태와 자료의 특성에 따르는 여러가지 효과적인 가시화방법들이 있다. 자료모임에는 스칼라값, 벡토르, 고차텐소르 또는 이런 자료형들이 여러가지로 결합되여 있을수 있다. 또한 자료모임은 2차원 또는 3차원적일수 있다. 색부호화는 자료모임을 가시화하는 한가지 방법이다. 보충적인 기술로서는 등값선도, 그라프와 도표, 면실감처리 그리고 체적내부의 가시화 등이 있다. 그리고 자료를 가시화하는데서 화상처리기술이 콤퓨터도형처리기술과 많이 결합된다.

수학자, 물리학자를 비롯하여 사람들은 가시화기술을 리용하여 수학함수와 과정을 분석하거나 또는 흥미 있는 도형적인 표현을 간단히 만들고 있다. 그림 1-51은 수학곡선함수의 색도면을 보여 주며 그림 1-52는 함수의 면표시를 보여 준다.

그림 1-51. 여러가지 색결합으로
그려 진 수학곡선

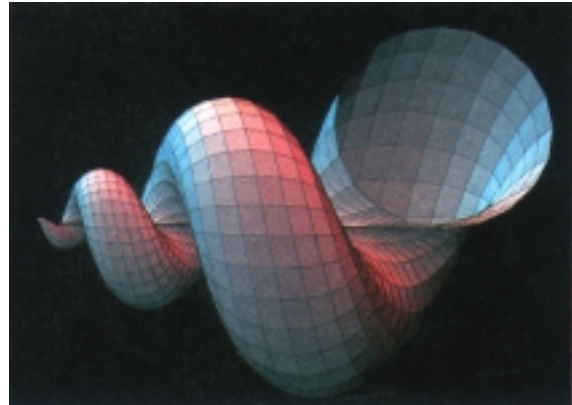



그림 1-52. 조명효과와 면실감처리기술을
적용하여 만든 3 차원함수의 면표시

그림 1-53에 보여 준 물체는 4 원수를 리용하는 프락탈절차로 만들어 낸것이고 그림 1-54는 위상수학적인 구조를 현시한것이다. 과학자들은 일반적인 자료모임을 가시화하는 방법들도 개발하고 있다. 그림 1-55 는 구면우에 분포된 자료를 가시화하는 일반적인 수법을 보여 주었다.

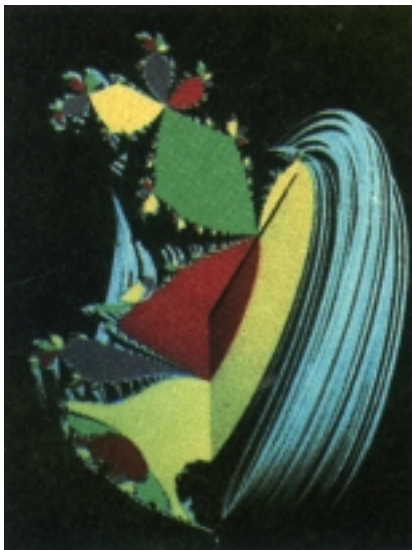



그림 1-53. 4 차원물체를 3 차원공간에 투영한 다음 다시 영상현시장치에 투영하고 색부호화하였다. 물체는 4 원수와 프락탈두제곱절차를 리용하여 만들었다. 복잡한 Julia 모임을 보여 주기 위하여 하나의 8 분공간은 뽑아 내였다.

그림 1-54. 3 차원유클리드공간에 투영된 3 구최소면(《달팽이》)의 실시간대화식 콤퓨터동화연구의 4 개 화면

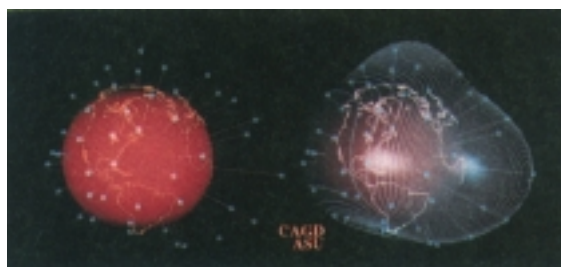



그림 1-55. 구면우에 분포된 자료를
가시화하는 방법

기타 여러가지 가시화방법들에 대한 응용실례의 일부를 그림 1-56부터 1-69에 보여 주었다. 이 그림들에서는 우주왕복선의 결면에서의 기류, 우뢰비의 수값모형화, 금속에서 균렬전파의 연구, 비행기날개에서 류체밀도의 색부호화된 도면, 자료모임에 대한 단층면, 단백질모형화, 분자구조의 립체적인 보기, 대양바닥의 모형, 쿠웨이트석유화재모의, 대기오염연구, 강냉이성장연구, 유적의 복원, 자동차사고통계그라프를 보여 준다.

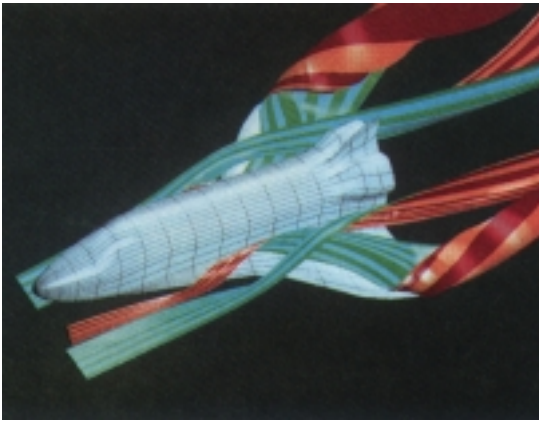



그림 1-56. 우주왕복선결면에서의 기류의 가시화

그림 1-57. 우뢰비안에서 기류의 수값모형

그림 1-58. 우뢰비면의 수값모형

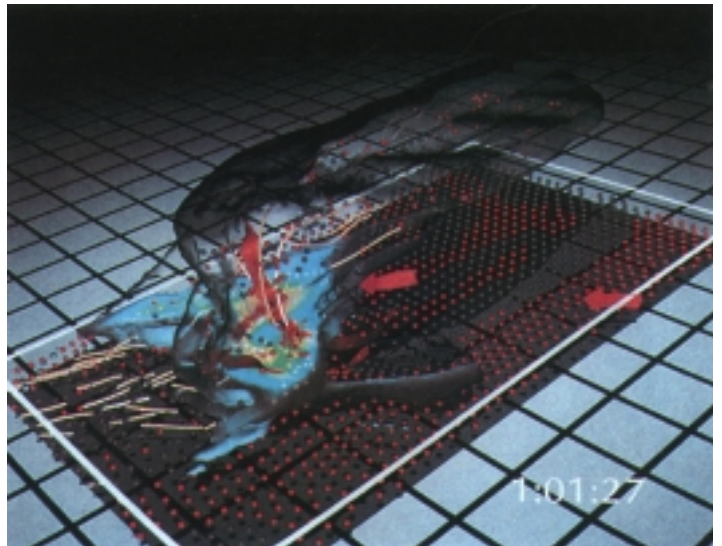
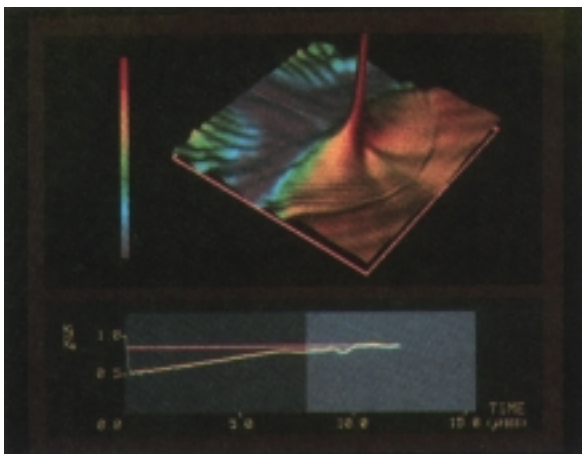



그림 1-59. 금속판의 균렬전파에 대한 연구에서 응력분포의 색부호화에 의한 가시화

그림 1-60. 류체의 동적모의(비행기날개주위의 격자면구간들에서 류체밀도를 색부호화한 그림이다.)

그림 1-61. 상품화된 층자르기쏘프트웨어(자료모임의 단층부에서 색부호화된 자료값을 보여 준다.)

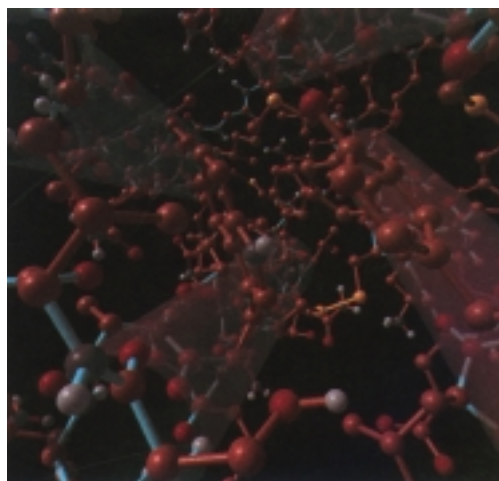



그림 1-62. 단백질구조의 가시화

그림 1-63.《boom》장치를 리용한 분자구조의 립체적인 보기

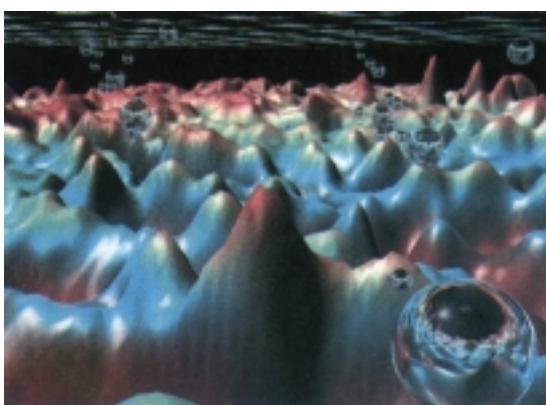



그림 1-64. 위성자료로부터 얻어 진 대양 바닥립체화상쌍의 하나의 화상

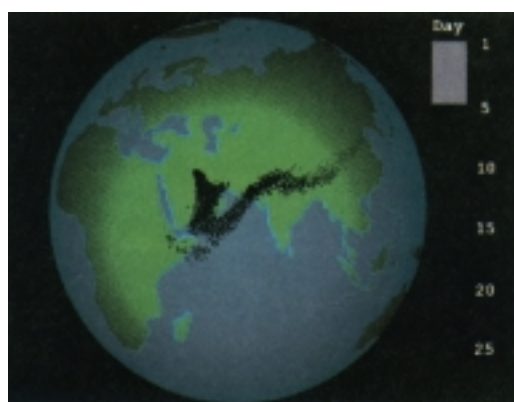



그림 1-65. 쿠웨이트석유화재 효과모의

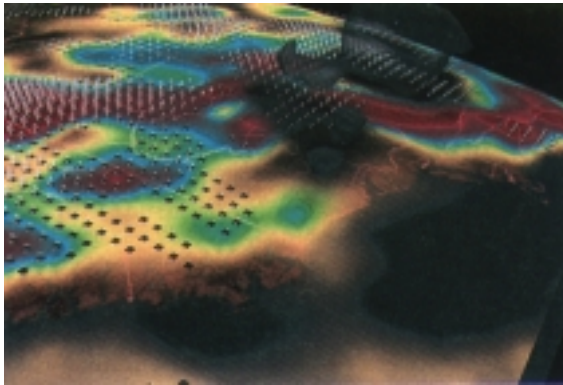



그림 1-66. 지구걸면에서의 오염구역의 가시화 (SO_4 오염은 푸른색으로, 산성비오염은 땅걸면 색으로, 비가 집중적으로 내리는 곳은 투명한 막대기로 표시하였다.)

그림 1-67. 강냉이이삭의 발육을 보여 주는 동화흐름의 한개 프레임

그림 1-68. Arizona 주의 Chaco Canyon 유적의 복원도

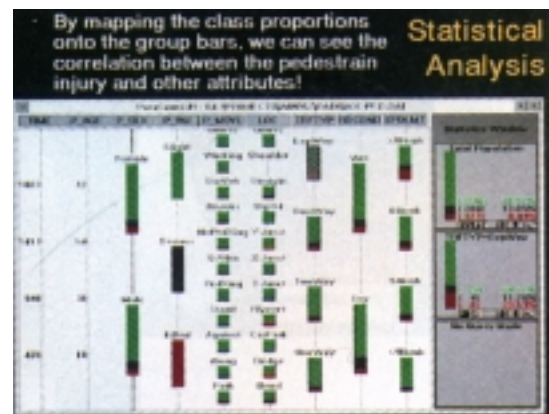



그림 1-69. 다차원표자료를 가시화하는 WinViz 라고 하는 견본기술(여기서는 자동차사고통계정보를 보행자와의 호상관계를 나타내는데 리용한다.)

7 절. 화상처리

비록 콤퓨터도형처리와 화상처리는 리용되는 방법들이 서로 겹치지만 두 분야는 근본적으로 다르다. 콤퓨터도형처리는 콤퓨터로 그림을 만드는데 리용된다. 그러나 **화상처리**기술은 사진이나 TV 화면과 같이 이미 존재하는 그림을 수정하거나 해석하는데 적용된다. 화상처리의 두가지 원리적인 응용은 화질개선과 로보트에서 리용되는것과 같은 시각정보의 기계적인식이다.

화상처리방법들을 적용하기 위하여서는 먼저 사진 또는 그림들을 화상파일로 수자화한다. 그다음 수자적인 방법으로 그림부분들을 재배치하거나 색간격을 늘이거나 명암을 개선한다. 화질을 높이는 화상처리방법의 응용실례를 그림 1-70 에 보여 주었다. 이런 기술들은 사진의 수정 및 재배치 기타 미술작업을 비롯하여 광고미술분야에 널리 리용된다. 이러한 방법들은 지구의 위성사진 및 은하수사

진을 분석하는데도 리용된다.

의학에서는 또한 화질을 높이는 화상처리기술들이 단층사진술과 수술조작모의에 널리 리용되고 있다. 단층사진술은 생리학적체계의 자름면상을 현시하는 X 선사진술이다. 콤퓨터 X 선단층촬영술과 양전자방사단층촬영술은 다같이 수자자료로부터 자름면을 복원하기 위하여 투영방법을 리용한다. 이런 기술들은 또한 내부기능을 감시하고 수술시 자름면을 보여 주는데도 리용된다. 다른 의학화상기술로서는 초음파 및 핵의학화상스캐너를 들수 있다. 초음파장치에서는 X 선대신에 초음파가 수자자료를 발생시키는데 리용되며 핵의학화상스캐너는 섭취된 방사성동위원소에서 나오는 방사선으로부터 수자자료를 수집하여 색부호화된 화상을 만든다.

일반적으로 화상처리와 콤퓨터도형처리는 많은 응용분야들에서 결합된다. 실례로 의학에서는 이 기술들을 물리적기능의 모형화 및 연구, 인공손발의 설계, 수술의 설계와 실천에 리용한다. 수술의 설계와 실천에서의 응용을 일반적으로 콤퓨터지원수술이라고 한다. 화상처리기술을 리용하여 몸체의 2차원자름면을 얻은 다음에 콤퓨터도형처리방법들을 리용하여 그것을 보면서 실지수술절차를 모방한 서로 다른 수술절단을 해본다. 이런 의학응용의 실례를 그림 1-71 과 그림 1-72 에 보여 주었다.

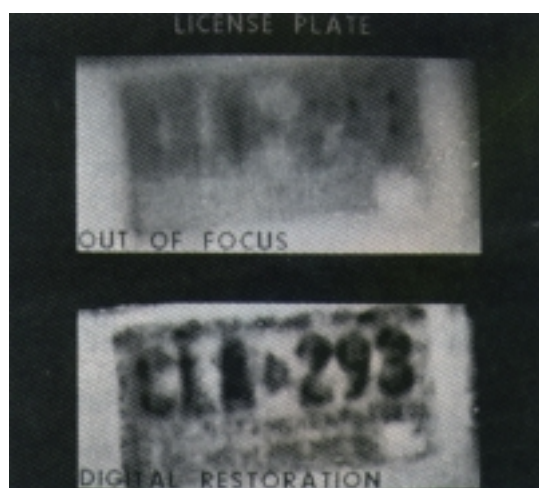



그림 1-70. 승인번호판의 희미한 사진이 화상처리후 명료해 졌다.

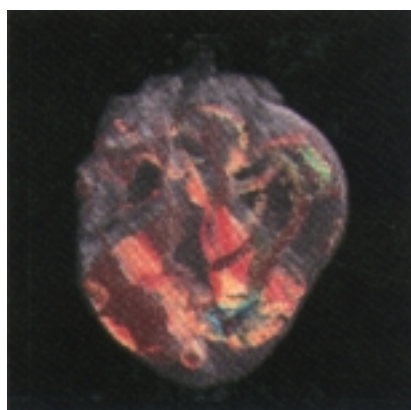



그림 1-71. 개의 심장구역안의 심장전기활성화준위를 반투명체적 실감처리로 가시화한 콤퓨터 동화의 한개 프레임

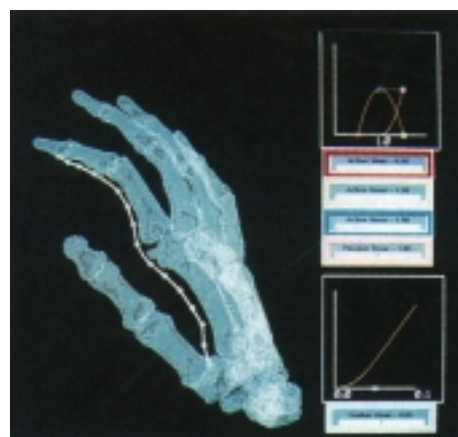



그림 1-72. 사람손의 뼈를 보여 주는 립체화상쌍의 한개 화상(CT 주사입력장치에서 얻은 자료로부터 실감처리된 이 화상은 재건수술에서 있을수 있는 힘줄경로를 보여 준다.)

8 절. 도형사용자대면부

지금 쏘프트웨어패키지들에서는 모두 **도형대면부**를 주고 있다. 도형대면부의 중요한 요소는 사용자가 여러 창문을 현시할수 있게 하는 창문관리자이다. 매개 창문에서는 도형적이거나 비도형적인 화면에 대한 여러가지 처리를 진행할수 있다. 매개 창문을 능동으로 만들기 위하여서는 대화식지시장치를 리용하여 그 창문안에서 간단히 찰칵하면 된다.

또한 대면부는 차림표와 그림기호를 현시하여 처리조작이나 파라메터값을 빨리 선택할수 있게 한다. 그림기호(icon)는 그것이 표현하는 처리항목과 같이 보이도록 그려 진 도형기호이다. 그림기호의 우점은 글로 서술하는것보다 화면공간을 적게 차지하며 잘 설계되면 더 빨리 리해될수 있다는것이다. 차림표는 본문서술과 그림기호들의 목록을 가지고 있다.

그림 1-73에서는 창문관리자, 차림표화면, 그림기호들을 가지고 있는 대표적인 도형대면부를 보여주었다. 이 실례의 차림표에서는 처리항목, 색값, 도형처리파라메터들을 선택한다. 그림기호는 색칠, 작도, 확대축소, 본문렬타자에 대한 항목들과 그림그리기와 관계되는 기타 다른 조작들을 표현한다.

그림 1-73. 다중창문구역, 차림표, 그림기호를 보여 주는 도형 사용자대면부

2장. 도형처리체계개론

지금 모든 분야들에서 콤퓨터도형처리의 능력과 쓸모가 실제적으로 광범히 인정됨으로써 넓은 범위에서 도형처리하드웨어 및 쏘프트웨어체계들을 사용할수 있게 되였다. 손바닥크기의 수산기를 비롯하여 수많은 일반목적콤퓨터들에는 2차원 및 3차원도형처리를 진행할수 있는 능력이 일반적으로 다 갖추어 져 있고 개인용콤퓨터들에서는 여러가지 대화식입력장치들과 도형처리쏘프트웨어들을 널리 리용할수 있게 되였다. 고급한 응용에서는 정교한 여러가지 전문적인 도형처리하드웨어체계들과 기술들을 선택하여 쓸수 있다. 이 장에서는 도형처리하드웨어요소들과 쏘프트웨어패키지들의 기본적인 특징을 고찰한다.

1절. 영상현시장치

일반적으로 도형처리체계에서 가장 기본적인 출력장치는 영상현시장치이다(그림 2-1). 대부분의 영상현시장치의 동작은 표준음극선관 CRT(Cathode-Ray Tube 의 략자)에 기초하고 있다. 하지만 여러가지 다른 기술들도 있으며 앞으로는 고체상태현시장치가 지배적일것이다.

그림 2-1. 콤퓨터도형처리워크스테이션

재생음극선관

그림 2-2 에서는 CRT의 기본동작을 설명한다. 전자총으로부터 방출되는 전자속(음극선)은 형광면의 지정된 위치로 향하게 하는 집초 및 편향체계를 통과한다. 이때 형광체는 전자속이 떨어 지는 매개 위치에서 작은 점빛을 낸다. 형광체가 내보내는 빛은 대단히 빨리 사라지기때문에 화면에서 그림을 유지하기 위한 어떤 방법이 필요하다. 형광체가 계속 빛을 내게 하는 한가지 방법은 전자속을 같은 점에로 빨리 다시 향하게 하여 그림을 반복적으로 다시 그리는것이다. 이런 형의 현시장치를 **재생 CRT**라고 한다.

CRT에서 전자총의 중요구성요소는 가열된 금속음극과 조종격자이다(그림2-3). 원기둥음극구조안의 가열선조에 전류를 흘려 음극에 열을 공급한다. 이것은 전자들로 하여금 뜨거운 음극면밖으로 나오게 한다. CRT관구안의 진공속에서 부로 대전된 자유로운 전자들은 그다음 높은 정전압을 띤 형광체쪽으로 가속된다. 가속전압은 형광면가까이의 CRT관구내부에 있는 정으로 대전된 금속면이나 그림 2-3에서와 같이 가속양극에 의하여 주어 진다. 일부 경우 가속양극과 집초체계를 하나의 구조단위(Unit)안에 넣기도 한다.

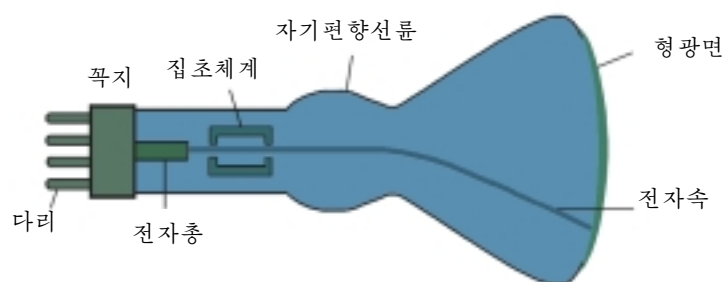



그림 2-2. 자기편향CRT의 기본구조

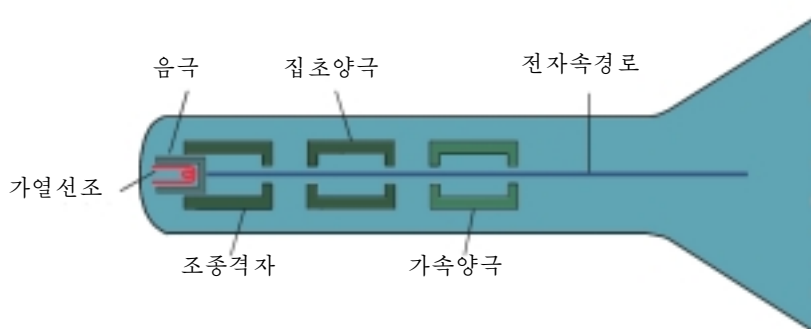



그림 2-3. 가속양극을 가지는 전자총의 동작

전자속의 세기는 조종격자에 가해 진 전압준위에 의하여 조종된다. 조종격자에 높은 부의 전압을 가하면 전자들은 배척되여 조종격자구조체의 끝에 있는 작은 구멍을 지나가지 못하게 됨으로써 속이 차단되게 된다. 조종격자에 걸리는 상대적으로 작은 부의 전압은 단순히 통과하는 전자들의 수를 감소시킨다. 형광체로부터 나오는 빛량은 화면을 때리는 전자들의 수에 관계되므로 조종격자전압을 변화시켜 현시의 밝기를 조종한다. 앞으로 제3장에서 설명하는바와 같이 개별적인 화면위치에서의 세기준위는 도형처리쏘프트웨어지령으로 지적한다.

CRT에서는 집초체계를 리용하여 전자속이 형광체를 때릴 때 한점에 집중되도록 한다. 한편 전자들은 서로 배척하므로 속은 화면에 다가가면서 퍼지게 된다. 집초는 전기 또는 자기마당에 의해 이루어 진다. 텔레비죤과 콤퓨터도형현시장치에서는 정전기적집초를 쓴다. 그림 2-3에서 보여 준바와 같이 정전기적집초에 의하여 전자속은 정전기적렌즈를 형성하는 정으로 대전된 금속원기둥을 통과한다. 정전기적렌즈의 작용은 광학렌즈가 빛속을 초점거리에 집중시키는것과 완전히 꼭 같은 방법으로 전자속을 화면의 중심에 집중시킨다. 이러한 렌즈집초효과는 CRT관구바깥주변에 설치된 선륜에 의하여 만들어 지는 자기마당에 의해서도 얻을수 있다. 자기렌즈집초는 화면에 최소크기의 점을 만들며 특별한 목적의 장치들에 리용된다.

정밀도가 높은 체계들에서는 화면의 모든 위치에서 전자속이 집중되도록 하는 보충적인 집초하드웨어를 리용한다. 대부분의 CRT에서는 만곡반경이 집초체계로부터 화면중심까지의 거리보다 더 크기때문에 전자속이 화면의 서로 다른 점에로 움직일 때 거리가 서로 다르다. 따라서 전자속은 화면의 중심에서만 정확히 집중되며 속이 화면의 바깥변두리로 움직일 때 현시되는 화상은 뿌옇게 된다. 이것을 보상하기 위하여 체계가 화면위치에 따르는 속의 집초를 조정하게 할수 있다.

집초와 마찬가지로 전자속의 편향도 전기마당 또는 자기마당에 의하여 조종된다. 음극선관은 현재 일반적으로 그림 2-2에서 설명되는바와 같이 CRT관구바깥에 설치되는 자기편향선륜에 의하여 만들어 진다. 두쌍의 선륜을 리용하는데 매 쌍의 선륜들은 CRT관구목의 반대쪽에 설치된다. 한쌍은 목

의 우와 아래에 설치되며 다른 쌍은 목의 반대쪽에 설치된다. 매개 선륜쌍에 의하여 만들어 지는 자기마당은 자기마당의 방향과 전자속이 움직이는 방향에 다같이 수직인 가로편향을 형성한다. 한쌍의 선륜은 수평편향을 형성하며 다른 쌍은 수직편향을 형성한다. 요구되는 편향량은 선륜으로 흐르는 전류를 조정하여 얻는다. 정전기적편향인 경우에는 CRT관구안에 두개의 평행금속판쌍을 설치한다. 한쌍의 금속판은 수직편향을 조종하기 위하여 수평으로 설치하며 다른 쌍은 수평편향을 조종하기 위하여 수직으로 설치한다(그림 2-4).

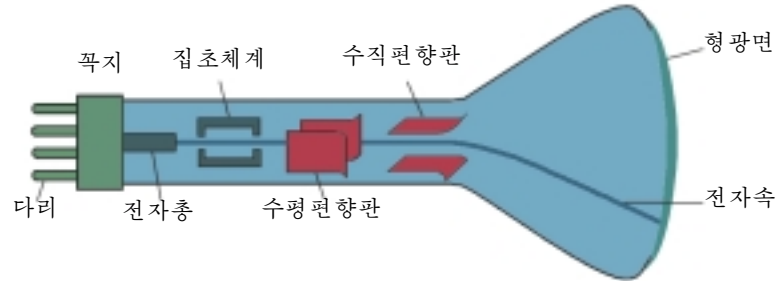



그림 2-4. CRT에서 전자속의 정전기적편향

화면에서 빛점은 CRT의 전자속에네르기가 형광체에 전달되여 발생한다. 전자속의 전자들이 형광체에 부딪칠 때 그의 운동에네르기는 형광체에 흡수된다. 속에네르기의 일부는 마찰에 의하여 열에네르기로 변환되고 나머지는 형광체원자안의 전자들을 보다 높은 량자에네르기준위에로 려기시킨다. 짧은 시간후에 《려기된》 형광체전자들은 자기의 안정한 기저상태로 다시 떨어 지면서 여분의 에네르기를 작은 빛량자에네르기로 내보낸다. 우리가 화면에서 보는것은 이 모든 전자빛방출의 결합된 효과이다. 발광점은 려기된 형광체전자들이 모두 자기의 기저에네르기준위에로 돌아 간후 인차 희미해 진다. 형광체에 의하여 방출되는 빛의 주파수(즉 색)는 려기된 량자상태와 기저상태사이의 에네르기차에 비례한다.

CRT에서는 각이한 종류의 형광체들을 리용할수 있다. 형광체들사이의 중요한 차이는 색이외에 CRT에서 속이 제거된후 얼마나 오래동안 빛방출을 계속하는가(즉 려기된 전자들을 얼마만한 시간동안 기저상태로 돌려 보내는가)하는 **지속성**이다. 지속성은 화면에서 방출되는 빛이 자기 본래 세기의 10분의 1로 감쇠되는데 걸리는 시간으로 정의한다. 지속성이 작은 형광체는 화면에서 깜빡임없이 그림을 유지하자면 재생속도가 높아야 한다. 지속성이 작은 형광체는 동화에 쓰이며 지속성이 큰 형광체는 대단히 복잡한 정적그림을 현시하는데 쓰인다. 1s이상의 지속성을 가지는 형광체들도 일부 있지만 도형현시장치에서는 지속성이 보통 10∼60ms범위인 형광체를 쓴다.

그림 2-5에서는 화면우의 점의 세기분포를 보여 주었다. 세기는 점의 중심에서 제일 크며 점의 변두리로 가면서 가우스분포에 따라 감소된다. 이 분포는 CRT전자속의 자름면의 전자밀도분포에 대응된다.

CRT에서 겹침이 없이 현시할수 있는 점의 최대수를 **분해능**이라고 한다. 분해능의 보다 정확한 정의는 수평 및 수직방향으로 찍을수 있는 cm당 점의 개수이다. 하지만 그것은 흔히 간단하게 매 방향에서의 점의 총수로 지적한다. 점의 세기는 가우스분포를 가지며(그림 2-5) 따라서 린접한 두 점사이의 거리를 중심세기의 약 60%가 되는 직경보다도 더 크게 취하면 두점을 구별하여 볼수 있다. 이 겹침위치를 그림 2-6에 보여 주었다. 점의 크기는 또한 세기에도 관계된다. 초당 더 많은 전자들이 형광체쪽으로 가속될수록 CRT속직경과 발광점이 커진다. 게다가 증가된 려기에네르기가 속의 경로에 잇닿은 형광체원자들에 간접적으로 뻗치게 되며 이것은 점의 직경을 증가시킨다. 그러므로 CRT분해

능은 형광체의 형, 현시되는 세기, 집초 및 편향체계에 관계된다. 고화질체계에서는 일반적으로 분해능이 1280×1024인데 더 높은 분해능도 사용되고 있다. 고분해능체계들을 고해상도체계라고 한다. 도형현시장치의 크기는 화면대각선의 길이로 규정하는데 약 12in부터 27in이다. 그이상의것도 있다. CRT현시장치는 여러가지 콤퓨터체계에 련결되므로 실지로 찍을수 있는 화면의 점의 수는 접속되는 체계의 능력에 관계된다.

그림 2-5. CRT 화면에서 발광하는 형광체점의 세기분포

그림 2-6. 발광하는 두 형광체점사이의 거리를 최대세기의 60%로 되는 직경보다 더 크게 취하면 두 점을 구별하여 볼수 있다.

영상현시장치의 다른 파라메터는 **종횡비**이다. 이 수자는 화면의 두 방향에서 같은 길이의 직선을 만드는데 필요한 수평점개수에 대한 수직점개수의 비이다(때때로 종횡비는 화면의 수직점에 대한 수평점의 비로 규정 짓는다). 종횡비 3/4 은 3 개 점으로 찍혀 지는 수직선이 4 개 점으로 찍혀 지는 수평선과 같은 길이를 가진다는것을 의미한다.

라스터주사현시장치

CRT를 쓰는 도형현시장치의 가장 일반적인 형은 텔레비죤기술에 기초한 **라스터주사**현시장치이다. 라스터주사체계에서 전자속은 우에서 아래로 한번에 한행씩 화면을 가로질러 훑는다. 전자속이 매개행을 지나갈 때 속의 세기는 발광점들의 무늬를 만들기 위하여 켜지고 꺼지고 한다. 그림의 정의는 **재생완충기** 또는 **프레임완충기**라고 하는 기억구역에 기억된다. 이 기억구역은 화면의 모든 점에 대한 세기값의 모임을 가지고 있다. 기억된 세기값은 그다음에 재생완충기로부터 다시 꺼내여 한번에 한행(주사선)씩 화면에 칠한다(그림 2-7). 화면의 매개 점은 **화소**(**픽셀**(pixel) 또는 **펠**(pel, picture element)의 줄임형)라고 한다. 라스터주사체계는 화면의 매개 점에 대한 세기정보를 기억하고 있으므로 미세한 명암과 색무늬를 포함하고 있는 장면의 현실적인 현시에 아주 적합하다. 가정용텔레비죤수상기와 인쇄기는 라스터주사방법을 리용하는 체계의 다른 실례들이다.

매 위치에서의 화소세기의 범위는 라스터체계의 능력에 관계된다. 간단한 흑백체계에서는 매개 화소점이 단지 켜지거나 꺼질뿐이므로 화면위치들의 세기를 조종하는데는 화소당 1bit만 필요하다. 2값체계에서 비트값 1은 전자속이 그 위치에서 켜진다는것을 가리키고 값 0은 속이 꺼진다는것을 의미한다. 색과 세기의 변화가 현시되게 하려면 추가적인 비트가 필요하다. 고급한 체계들에서는 화소당 24bit까지도 할당하는데 그러면 프레임완충기의 기억용량이 수MB로 될수 있다(체계의 분해능에 관계된다.). 화소당 24bit 그리고 화면분해능이 1024×1024인 체계에서는 프레임완충기에 3Mbit의 기억기가 요구된다. 화소당 1bit를 가지는 흑백체계에서는 프레임완충기를 일반적으로 **비트배렬**(bitmap)이라고 하며 화소당 여러 비트를 가지는 체계에서는 프레임완충기를 흔히 **점배렬**(pixmap)이라고 한다.

라스터주사현시장치에서의 재생은 보통 초당 60~80프레임의 속도이지만 일부 체계들은 재생속도가 더 빠르다. 재생속도를 초당 싸이클 또는 헤르쯔(Hz)로 표현한다. 여기서 한싸이클은 한프레임에 대응한다. 이 단위를 리용하면 초당 60프레임의 재생속도를 간단히 60Hz로 표현할수 있다. 매개 주사선의 마지막에서 전자속은 다음 주사선의 현시를 시작하기 위하여 화면의 왼쪽으로 돌아 간다. 재생후 매개 주사선의 화면왼쪽에로의 돌리기를 전자속의 수평귀선이라고 한다. 1/80~1/60s로 현시되는

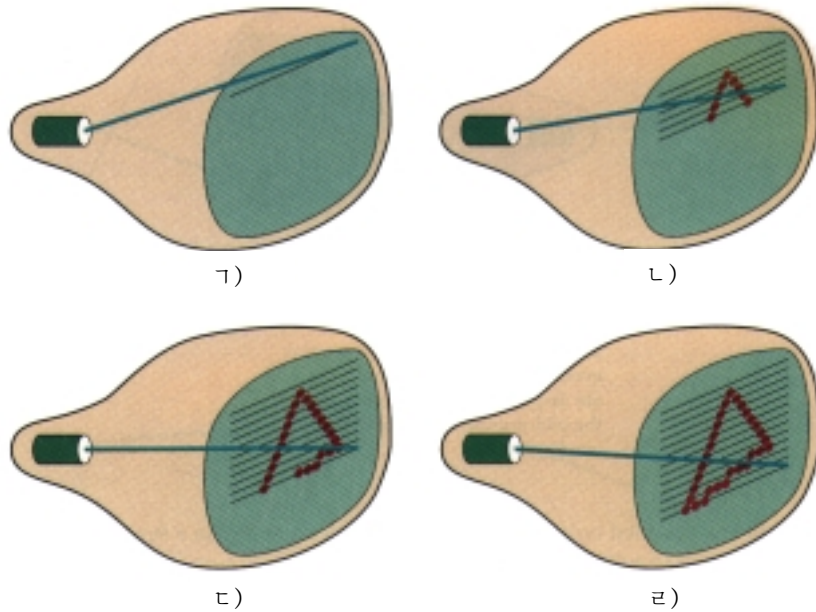



그림 2-7. 라스터주사체계는 매개 주사선을 따라 놓이는 불련속점들의 모임으로 물체를 현시한다.

매개 프레임의 마지막에 전자속은 다음프레임을 시작하기 위하여 화면의 왼쪽 웃구석에로 돌아 간다(수직귀선).

일부 라스터주사체계(또는 TV수상기)에서는 매개 프레임을 두번의 주사로 현시한다. 첫 통과에서 속은 우에서부터 주사선을 하나건너씩 훑는다. 다음 수직귀선후 속은 남은 주사선들을 훑는다(그림 2-8). 이 경우 우에서 아래로 단번에 모든 선들을 따라 훑는데 드는 시간의 절반시간에 현시된 전체 화면을 보게 한다. 이렇게 하면 무엇보다먼저 보다 느린 재생속도를 리용할 수 있다. 례를 들면 초당 30 프레임의 비간격식현시장치에서는 깜박임을 느낄수 있으나 간격식을 쓰면 매 통과가 1/60s에 이루어 지므로 초당 60프레임에 가까운 재생속도를 가져 오게 된다. 이것은 린접한 주사선들이 류사한 현시정보를 포함한다는 조건하에서 깜박임을 피하기 위한 효과적인 기술이다.

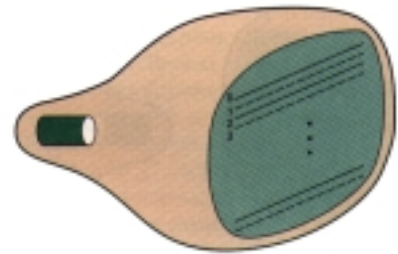



그림 2-8. 라스터주사현시장치에서 간격식주사 (먼저 짝수번호주사선(실선)우의 모든 점이 현시되고 다음에 홀수번호주사선(점선)우의 모든 점이 현시된다.)

우연주사현시장치

우연주사현시장치로 동작할 때 CRT에서는 전자속이 그림이 그려 지는 부분으로만 향한다. 우연주사현시장치는 그림을 한번에 한개 선씩 그린다. 그러므로 **벡토르현시장치**(또는 획쓰기, 또는 **필순현시장치**)라고도 한다. 우연주사체계에서는 그림의 요소선들을 임의로 지적되는 순서로 그리고 재생시킬수 있다(그림 2-9). 펜작도기는 이런 방법으로 동작하는 우연주사경복사장치의 한 실례이다.

우연주사체계에서 재생속도는 현시되는 선의 수에 관계된다. 이때 그림의 정의는 **재생현시파일**이라고 하는 기억구역에 선그리기명령들의 모임으로 기억시킨다. 때때로 재생현시파일을 **현시목록, 현시프로그람** 또는 간단히 **재생완충기**라고 한다. 체계는 현시파일에 있는 명령모임을 순환하면서 매개 요소선들을 차례로 그려 지적된 그림을 현시한다. 모든 선그리기명령이 다 수행된후 체계는 목록안의

첫번째 명령에로 되돌아 와 반복한다. 우연주사현시장치는 그림의 모든 요소선들을 초당 30～60번 그리게 설계된다. 고급한 벡토르체계는 이 재생속도로 약 10만개의 《짧은》선을 처리할수 있다. 선들의 모임이 작을 때에는 재생속도가 초당 60프레임을 넘지 않도록 하기 위하여 매개 재생주기를 지연시킨다. 왜냐하면 선모임의 빠른 재생은 형광체를 타게 할수 있다.

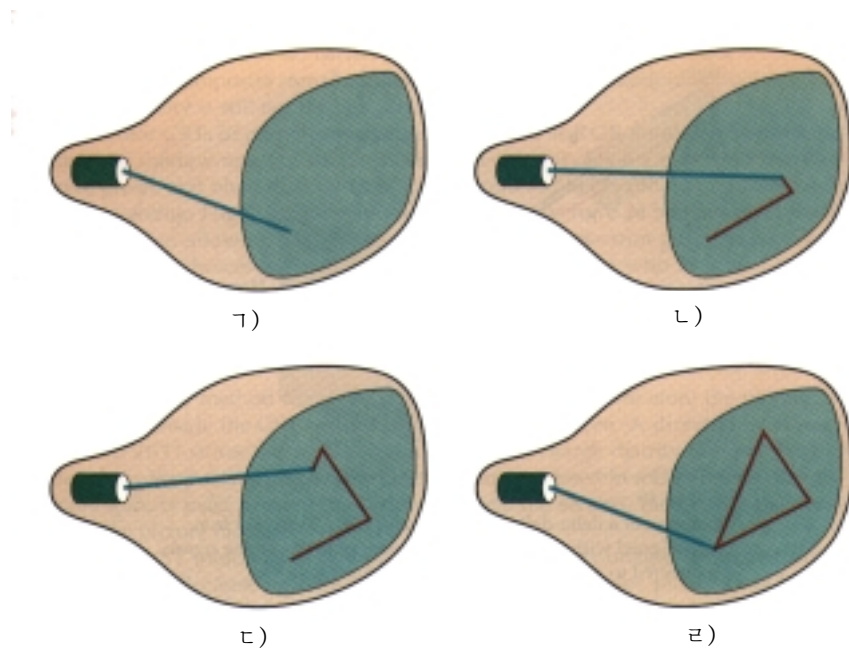



그림 2-9. 우연주사체계는 임의로 지적되는 순서로 물체의 요소선들을 그린다.

우연주사체계는 선그리기를 위하여 설계되였으므로 실제적인 명암장면은 현시할수 없다. 그림의 정의는 화면의 모든 점들에 대한 세기값들의 모임이 아니라 선그리기명령모임으로 기억되기때문에 벡토르현시장치는 일반적으로 라스터체계보다 분해능이 높다. 또한 벡토르현시장치는 CRT속이 직접 선경로를 따르기때문에 매끈한 선그림을 만든다. 반대로 라스터체계는 불련속점들의 모임으로 찍혀지는 터실터실한 선을 만든다.

색CRT현시장치

CRT현시장치는 서로 다른 색갈의 빛을 내보내는 형광체들의 조합을 써서 색그림을 현시한다. 서로 다른 형광체로부터 방출되는 빛을 조합하면 여러가지 색갈을 만들수 있다. CRT로 색현시장치를 만드는 방법에는 기본적으로 두가지가 있는데 전자속투과법과 그림자마스크법이다.

색그림을 현시하기 위한 **전자속투과법**은 우연주사현시장치에 리용되였다. 보통 붉은색 및 풀색의 두개의 형광체층을 CRT내부에 칠한다. 현시되는 색은 전자속이 형광체층안으로 얼마나 깊이 침투하는가에 관계된다. 느린 전자속은 바깥의 붉은색층만을 려기시킨다. 빠른 전자속은 붉은색층을 투과하여 내부의 풀색층을 려기시킨다. 중간속도의 속에서는 두가지의 보충적인 색 즉 감색과 누런색을 나타내는 붉은색과 풀색빛의 결합이 나온다. 전자들의 속도 즉 화면의 임의의 점에서의 색은 전자속의 가속전압에 의하여 조종된다. 속투과법은 우연주사현시장치에서 색을 만드는 값 눅은 방법이지만 네가지 색만 낼수 있으며 다른 방법들보다 화질이 좋지 못하다.

그림자마스크법은 속투과법보다 더 넓은 범위의 색을 만들기때문에 색TV를 포함하여 라스터주사체계에서 공통적으로 리용된다. 그림자마스크 CRT는 매개 화소위치에서 3개의 형광체색점을 가진다.

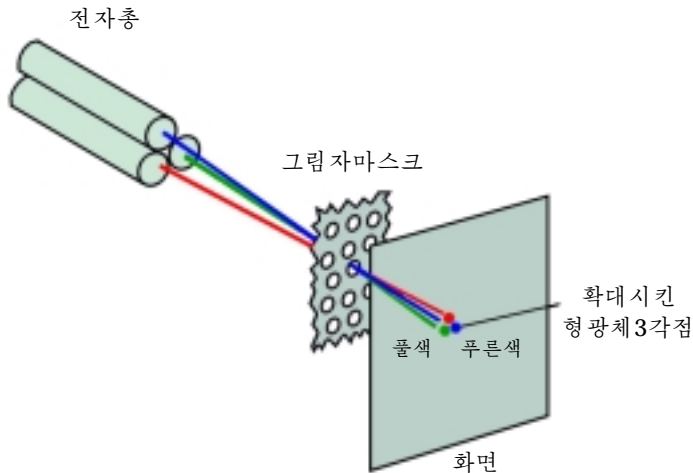



그림 2-10. 델타-델타그림자마스크 CRT의 동작(화면우의 3각색점모양으로 배렬된 3개의 전자총은 그림자마스크에 의하여 매개 3각점에로 향해 진다.)

한개의 형광체점은 붉은색빛을 내보내고 다른것은 풀색 그리고 또 다른것은 푸른색빛을 내보낸다. 이 형의 CRT는 매 색점에 하나씩 3개의 전자총을 가지고 있으며 형광면뒤에 그림자마스크를 가진다. 그림 2-10에서는 색 CRT체계에서 공통적으로 리용되고 있는 델타-델타그림자마스크법을 보여 주었다. 3개의 전자속은 그림자마스크에 한덩어리로 편향 및 집초된다. 그림자마스크는 형광체점들모양으로 배렬된 일련의 구멍들을 가지고 있다.

3개의 속은 그림자마스크에 있는 구멍을 통과하여 형광체3각점들을 활성화하는데 이것은 화면에 작은 색점으로 나타난다. 3각형안의 형광체점들은 전자속이 그림자마스크를 통과할 때 대응하는 색점만 활성화될수 있도록 배렬되여 있다. 3개 전자총의 다른 구성안은 3개의 전자총을 직선적으로 배렬하는것이다. 이때 화면우의 붉은색-풀색-푸른색점들은 3각형모양대신에 주사선을 따라 배렬된다. 전자총의 이런 직선배렬은 배렬이 쉬워 고분해능색CRT에서 리용되고 있다.

3개 전자속의 세기를 변화시켜 그림자마스크CRT에서 색변화를 얻는다. 붉은색 및 풀색총를 끄면 푸른색형광체로부터 나오는 색만을 얻게 된다. 전자속세기의 서로 다른 조합으로 매개 화소위치에서 서로 다른 색갈의 작은 빛점을 만드는데 그것은 우리의 눈이 세가지 색을 하나로 합성하는 성질이 있기때문이다. 보이는 색은 붉은색, 풀색, 푸른색형광체들의 려기량에 관계된다. 백색(또는 회색)령역은 세점이 같은 세기로 활성화된 결과이다. 누런색은 풀색과 붉은색점만으로 만들어 지며 분홍색은 푸른색과 붉은색점으로 만들어 진다. 그리고 푸른풀색은 푸른색과 풀색이 꼭같이 활성화될 때 나타난다. 일부 저가격체계에서는 전자속을 켜기/끄기(on/off)만으로 설정할수 있어 8가지 현시색으로 제한된다. 보다 고급한 체계들에서는 전자속에 대하여 중간세기준위를 설정할수 있어 수백만개의 서로 다른 색을 발생시킬수 있다.

색도형처리체계는 여러가지 형의 CRT현시장치를 쓸수 있도록 설계된다. 일부 저가격가정용콤퓨터체계와 비데오놀이감들은 색TV수상기와 RF(Radio-frequency)변조기를 쓰게끔 설계되였다. RF변조기의 목적은 TV방송국으로부터의 신호를 모의하는것이다. 이것은 그림의 색 및 세기정보를 TV가 입력하는데 필요한 방송주파수의 반송파신호에 태운다는것을 의미한다. 그러면 TV안의 전자회로가 RF변조기로부터 이 신호를 잡아 그림정보를 뽑아 내여 그것을 화면에 그린다. 짐작할수 있는바와 같이 RF변조기와 TV회로에 의한 그림정보의 이 추가적인 처리는 현시되는 화상의 질을 낮춘다.

합성현시장치(composite monitor)는 방송회로(고주파회로)를 지나갈수 있게 만든 TV수상기와의 결합기이다. 이 현시장치는 반송파신호는 필요 없지만 여전히 그림정보가 결합될것을 요구한다. 그림정보는 합성신호에 결합되며 그다음 현시장치에서 분리된다. 그림의 질은 여전히 제일 좋은것은 못된다.

도형처리체계에서 색CRT들은 **RGB현시장치**(RGB monitor)로 설계된다. 이 현시장치는 그림자마스크법을 리용하며 매개 전자총(붉은색, 풀색 및 푸른색)에 대한 세기준위를 중간처리없이 콤퓨터체계로부터 직접 잡는다. 고급한 라스터도형처리체계는 프레임완충기에서 화소당 24bit, 매개 전자총에서

256개의 전압준위를 설정하여 매개 화소에 대하여 1700만가지 색을 선택할수 있다. 화소당 24bit의 기억기를 가지는 RGB색체계를 일반적으로 **완전색체계**(full-color system) 또는 **자연색체계**(true-color system)라고 한다.

직시축적관

화면에서 화상을 유지하기 위한 또 하나의 방법은 화면의 재생대신에 CRT내부에 그림정보를 기억시키는것이다. **직시축적관**(direct-view storage tube)은 그림정보를 형광면뒤에 전하분포로 기억시킨다. 직시축적관에서는 두개의 전자총이 리용된다. 하나(primary gun)는 그림의 모양을 기억시키는데 리용되고 두번째(flood gun)는 현시그림을 유지한다.

DVST현시장치는 재생CRT와 비교할 때 우결함을 다같이 가지고 있다. 재생이 필요 없기때문에 대단히 복잡한 그림을 깜빡임없이 매우 높은 분해능으로 현시할수 있다. DVST체계의 결함은 대체로 색을 현시할수 없으며 그림의 선택된 부분을 지울수 없는것이다. 부분그림을 없애기 위해서는 전체화면을 지우고 수정된 그림을 다시 그려야 한다. 지우기 및 다시그리기처리는 복잡한 그림에서 수s 걸릴수 있다. 이런 리유로 기억식현시장치는 거의나 라스터체계로 교체되였다.

평판현시장치

비록 대부분의 도형현시장치들이 여전히 CRT로 만들어 지고 있지만 CRT현시장치를 대신하는 다른 기술들이 나오고 있다. **평판현시장치**(flat-panel display)란 말은 CRT에 비하여 체적, 무게, 전원요구를 줄인 영상장치들의 종류를 말하는것이다. 평판현시장치의 중요한 특징은 CRT보다 두께가 얇은것이며 그래서 그것을 벽에 걸거나 손목에 찰수 있다. 지어 일부 평판현시장치에는 글을 쓸수도 있기때문에 그것을 주머니용수첩으로도 쓸수 있다. 평판현시장치는 현재 소형TV현시장치, 수산기, 주머니용비데오놀이감, 휴대용콤퓨터, 비행기안에서의 의자걸이영화보기, 승강기안의 광고판, 든든하고 휴대할수 있는 현시장치를 요구하는 응용들에서 쓰이고 있다.

평판현시장치는 **방출형현시장치**(emissive display)와 **비방출형현시장치**(nonemissive display)로 나눈다. 방출형현시장치(또는 **방사기**(emitter))는 전기에네르기를 빛으로 변환하는 장치이다. 플라즈마평판, 박막전기발광현시장치, 발광2극소자들은 방출형현시장치의 실례들이다. 또한 평판식CRT들도 발명되였는데 여기서는 전자속들이 화면에 평행으로 가속된 다음에 화면에 90°로 편향된다. 그러나 평판식CRT들은 다른 방출형장치들만큼 성공적이라고 인정되지 못하였다. 비방출형현시장치(비방사기)는 해빛 또는 다른 빛원천으로부터의 빛을 도형무늬로 변환하는 광학효과를 리용한다. 비방출형평판현시장치의 가장 대표적인 실례는 액정장치이다.

플라즈마현시판(plasma panel)은 **기체방전현시장치**(gas-discharge display)라고도 하는데 두개의 유리판사이의 구역에 일반적으로 네온을 포함하는 기체혼합물을 채워 만든다. 한쪽 유리판우에는 여러개의 수직전도띠들이 놓이고 다른쪽 유리판에는 수평전도띠들이 설치된다(그림2-11). 수평

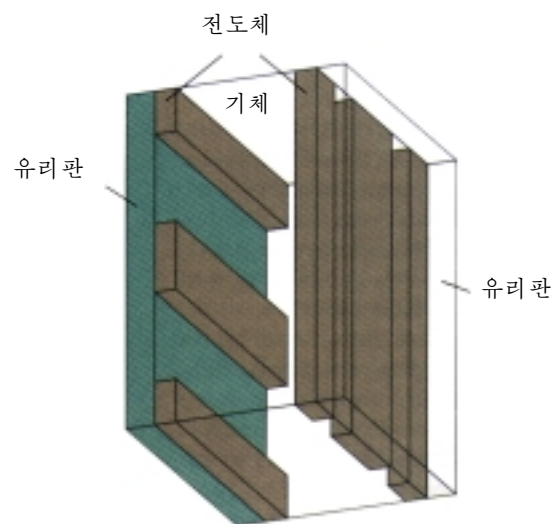



그림 2-11. 플라즈마평판현시장치의 기본구조

및 수직전도체쌍에 가해 진 점화전압은 두 전도체가 사귀는 곳에 있는 기체를 전자와 이온의 발광플라즈마로 만든다. 그림의 정의는 재생완충기에 기억되며 점화전압은 전도체들의 사귐부에 있는 화소위치를 재생하기 위하여 초당 60번 인가한다. 점화전압을 빨리 인가하여 보다 밝은 현시를 얻기 위하여 교번전류법(alternating-current method)이 리용되고 있다. 화소들사이는 전도체의 전기마당에 의하여 격리된다. 그림 2-12는 선명도가 높은 플라즈마현시판을 보여 준다. 플라즈마현시판의 한가지 결함은 그것이 엄격히 단색장치라는것인데 지금은 색 및 흑백계조를 현시할수 있는 체계들도 개발되고 있다.

박막전기발광현시장치(Thin-film electroluminescent display)는 구성에서 플라즈마현시판과 류사하다. 차이점은 유리판사이구역에 기체대신에 망간을 첨가한 류화아연과 같은 형광체를 채운것이다(그림 2-13). 사귀는 두 전극에 충분히 높은 전압을 가하면 형광체는 두 전극의 사귐구역에서 도체로 된다. 이때 전기에네르기는 플라즈마효과와 류사한 점빛을 내보낸다. 전기발광현시장치는 플라즈마현시판보다 더 많은 전력을 요구하며 질 좋은 색 및 흑백계조현시장치는 만들기 힘들다.

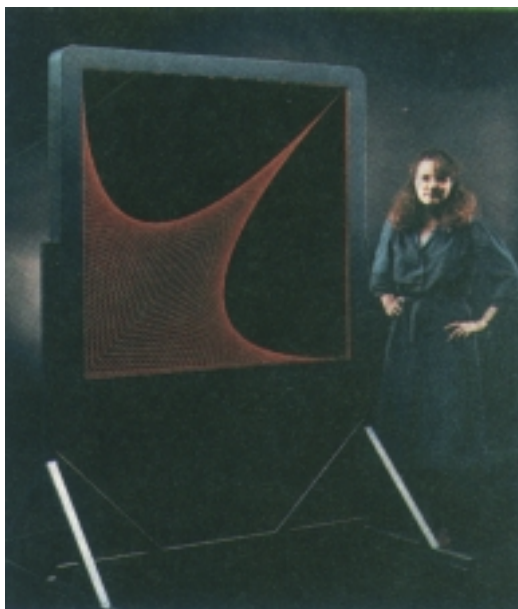



그림 2-12. 2048 × 2048 의 분해능과 1.5m 의 화면대각선을 가지는 플라즈마평판현시장치

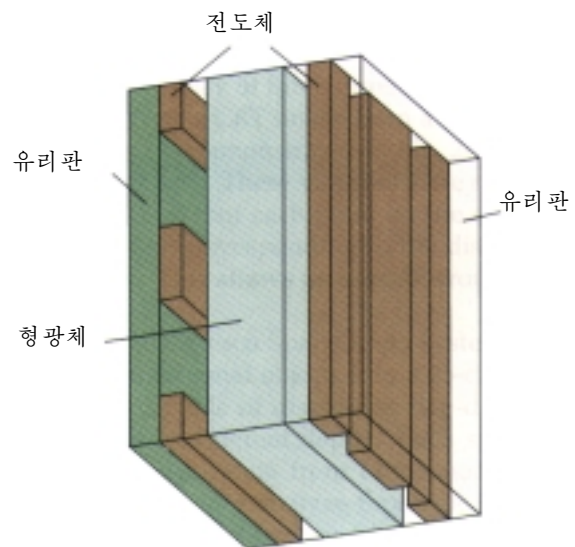



그림 2-13. 박막전기발광현시장치의 기본구조

세번째 형의 방출형장치는 **발광2극소자**(light-emitting diode, LED)이다. 현시장치에서 화소위치를 형성하기 위하여 2극소자행렬이 리용되며 그림의 정의는 재생완충기에 기억된다. CRT의 주사선재생에서와 같이 정보를 재생완충기에서 읽어 표시장치에 빛무늬를 만들기 위하여 2극소자에 가해 지는 전압준위로 변환된다.

액정현시장치(Liquid-crystal displays)는 수산기(그림 2-14)와 휴대용콤퓨터(그림 2-15)와 같은 작은 체계들에서 리용된다. 이 비방출형장치는 주위 또는 내부의 빛원천으로부터 편극된 빛을 액정재료로 통과시켜 그림을 만든다. 이 액정재료는 빛을 차단시키거나 투과할수 있도록 정렬시킬수 있다.

액정이라는 말은 이 화합물이 결정의 분자배렬을 가지면서도 액체와 같이 흐른다는 뜻을 나타낸다. 평판현시장치들은 일반적으로 네마틱(실 같은 모양)액정화합물을 사용하는데 이것은 가늘고 긴 막대기모양의 분자들의 긴축을 일직선으로 유지하는 경향성이 있다. 평판현시장치는 그림 2-16 에서 보여 준바와 같이 네마틱액정으로 만들수 있다. 각각 오른쪽 편광자를 가지고 있는 두개의 유리판사이에

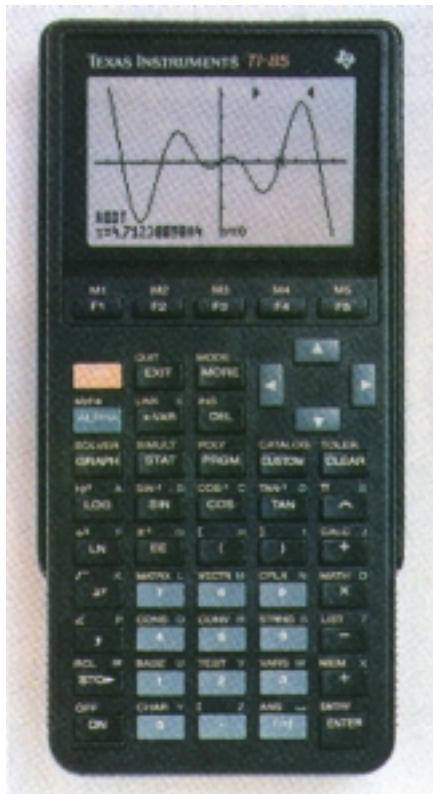



그림 2-14. LCD 화면을 가지는 수산기

그림 2-15. 256 색, 화면분해능 640 × 400, 화면대각선 9in 인 휴대용콤퓨터의 피동행렬액정현시장치

액정재료를 삽입한다. 수평방향의 투명전도체의 행들이 하나의 유리판에 설치되고 수직방향의 전도체의 렬들이 다른 유리판에 설치된다. 두 전도체의 사귐점은 화소위치를 결정한다. 정상상태에서 분자들은 그림 2-16에서 보여 주는바와 같이 열림상태로 정렬된다. 재료를 통과하는 편광빛은 반대쪽 편광자를 통과할수 있게 꼬여진다. 그러면 빛은 보는 사람에게 반사된다. 화소를 닫김상태로 절환하기 위하여서는 두개의 교차되는 전도체에 빛이 꼬이지 않도록 분자들을 정렬시키는 전압을 인가한다. 이런 형의 평판현시장치를 피동행렬LCD라고 한다. 그림의 정의는 재생완충기에 기억되며 화면은 방출형장치에서와 같이 초당 60프레임의 속도로 재생된다. 뒤조명도 역시 일반적으로 고체전자기구를 리용하므로 체계는 바깥광원에 전적으로 의존하지는 않는다. 서로 다른 재료

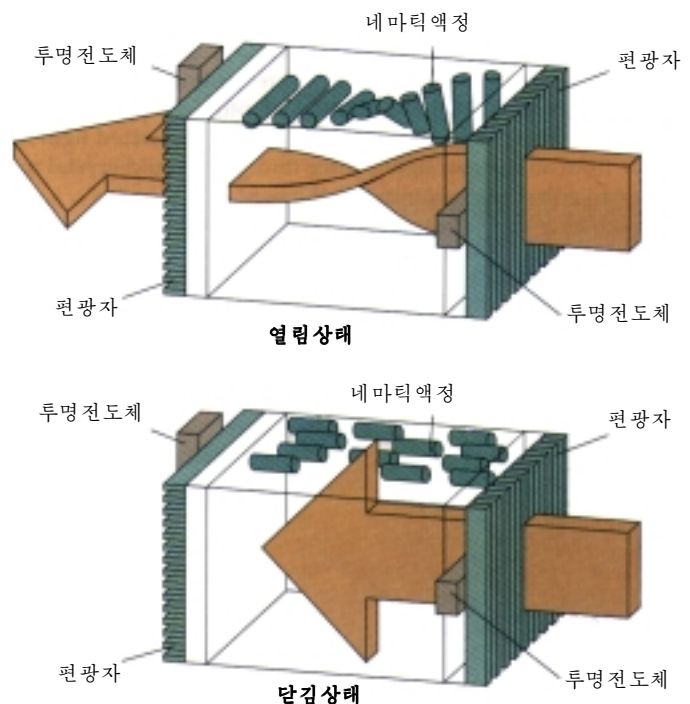



그림 2-16. 대부분의 액정현시장치설계에서 리용되는 빛꼬임덧문효과

나 물감을 리용하여 매개 화면위치에 세가지 색화소를 놓으면 색을 현시할수 있다. LCD를 만드는 다른 방법은 박막3극소자기술을 리용하여 매개 화소위치에 3극소자를 놓는것이다. 3극소자는 화소위치에서의 전압을 조종하며 액정세포로부터 전하들이 점차적으로 루설되는것을 방지하기 위하여 리용된다. 이런 장치를 능동행렬현시장치라고 한다.

3 차원적으로 보는 장치

진동식림성거울로부터 CRT화상을 반사시키는 기술을 리용하여 3차원장면현시를 진행하는 도형현시장치들이 발명되였다. 그림 2-17에 이런 체계의 동작을 보여 주었다. 가변초점거울이 진동하면 그것은 초점거리를 변화시킨다. 이 진동은 CRT에서의 물체현시와 동기를 맞추어 진행함으로써 물체의 매개 점이 지적된 보기위치로부터 그 점까지 거리에 대응하는 공간적인 위치에 거울반사되도록 한다. 이것은 우리가 물체나 장면주위를 걸으면서 여러가지 방향에서 그것을 본다는 감을 느끼게 한다.

그림 2-18에서는 Genisco의 SpaceGraph체계를 보여 주었다. 이것은 3차원물체를 25×25×25cm체적안에 투영하는 진동거울을 리용하고 있다. 이 체계는 또한 서로 다른 깊이에서 선택되는 물체의 2차원자름면의 얇은 쪼각을 현시할수 있다 이런 체계들은 의학에서 초음파장치 및 CAT주사장치로부터의 자료분석, 지질학에서 위상학적 및 지진자료분석, 립체를 다루는 설계, 분자 및 지형과 같은 체계의 3차원모의에서 리용된다.

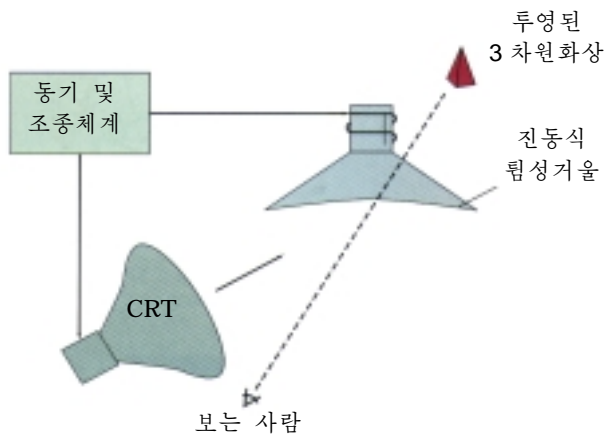



그림 2-17. 장면안의 점 깊이에 맞게 초점거리를 변화시키는 진동거울을 리용하는 3 차원현시체계의 동작

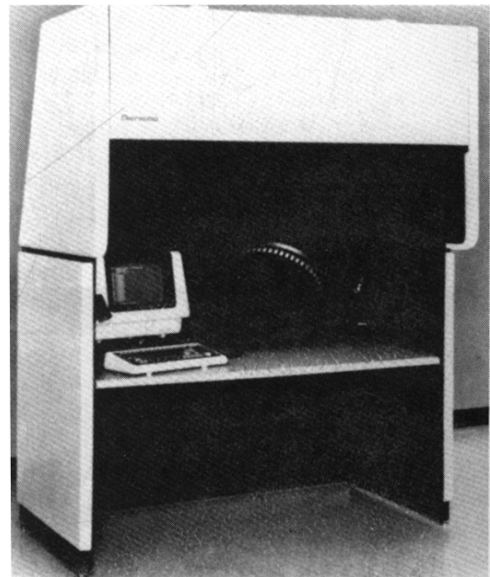



그림 2-18. SpaceGraph 대화식도형처리체계는 진동식림성거울을 리용하여 물체를 3 차원적으로 현시한다.

립체 및 가상현실체계

3차원물체를 표현하는 다른 기술은 립체상을 현시하는것이다. 이 방법은 진짜 3차원화상을 만들지는 않지만 장면이 깊이가 있게 보이도록 관측자의 매 눈에 서로 다른 상을 제시하는 방법으로 3차원효과를 준다(그림 2-19).

립체투영을 얻기 위하여서는 먼저 매 눈(왼쪽 및 오른쪽)에 대응한 시선방향에서 형성되는 장면의 두개 상을 얻어야 한다. 두개 상은 서로 다른 보기위치를 가지는 콤퓨터생성장면으로 만들수도 있고 일부 물체 또는 장면을 립체카메라를 리용하여 사진 찍을수도 있다. 왼쪽 눈으로 왼쪽 상을, 오른쪽 눈으로 오른쪽 상을 동시에 볼 때 두개 상은 하나의 화상으로 겹쳐 져 깊이를 가지는 장면으로 느껴 진다. 그림 2-20 에서는 립체투영을 얻기 위하여 콤퓨터로 생성한 장면의 두개 상을 보여 주었다. 보기에 편리하게 하기 위하여 한쪽 눈에만 보이는 이 장면의 왼쪽과 오른쪽 변두리구역을 없앴다.

그림 2-19. 립체투영을 보기

그림 2-20. 립체보기쌍

립체효과를 만드는 한가지 방법은 매 두개의 보임상을 라스터체계에서 재생주파수로 엇바꾸어 현시하는것이다. 화면은 안경을 통하여 보는데 매개 렌즈는 화면의 현시에 동기를 맞추어 하나의 쌍만 보이게 한다. 그림 2-21 에 액정덧문으로 만든 립체안경과 화면에서의 보임상과 안경을 동기시키는 적외선발진기를 보여 주었다.

그림 2-21. 립체장면을 보는 안경과 적외선동기신호발진기

립체보기는 또한 사용자가 장면안에 들어 갈수도 있고 환경과 호상작용할수도 있는 **가상현실**(virtual reality)체계의 구성요소이다. 립체상을 만드는 광학체계를 가진 **모자**(headset)(그림 2-22)는 일반적으로 장면안의 물체를 가리키고 조작하기 위한 대화식입력장치들과 함께 리용된다. 모자의 수감체계는 보는 사람의 위치자리길을 보존하며 따라서 보는 사람이 걸으면서 화면과 호상작용하고 물체의 앞과 뒤를 볼수 있게 한다. 그림 2-23은 가상현실환경과의 호상작용을 보여 주는데 모자와 오른쪽 손에 낀 자료장갑(2장 5절)을 리용하고 있다.

그림 2-22. 가상현실체계에서 리용되는 모자

그림 2-23. 가상현실환경과의 호상작용

그림 2-24. 립체안경과 함께 쓰이는 머리의 위치를 감시하기 위한 초음파추적장치

대화식가상현실환경은 모자대신에 립체안경과 영상현시장치로도 볼수 있다. 이것은 저가격가상현실체계를 얻는 방법이다. 실례로 그림 2-24 에서는 6개 자유도를 가지는 초음파추적장치를 보여 주었다. 추적장치는 영상현시장치우에서 머리의 움직임을 감시하여 머리위치가 변할 때 보기위치가 다른 장면이 얻어 지게 한다.

2절. 라스터주사체계

대화식라스터도형처리체계에는 여러가지 처리단들이 쓰인다. 현시장치의 동작을 조종하는데 중앙처리장치 CPU외에 **영상조종기**(video controller) 또는 **현시조종기**(display controller)라고 하는 전용처리기들이 리용된다. 그림 2-25에 간단한 라스터체계의 구성을 보여주었다. 여기서 프레임완충기는 체계기억기의 임의의 d위치에 있을수 있으며 영상조종기는 화면을 재생하기 위하여 프레임완충기를 호출한다. 보다 고급한 라스터체계들에서는 영상조종기외에 여러가지 도형처리조작을 진행하기 위하여 협동처리기 및 가속기와 같은 다른 처리기들도 쓴다.

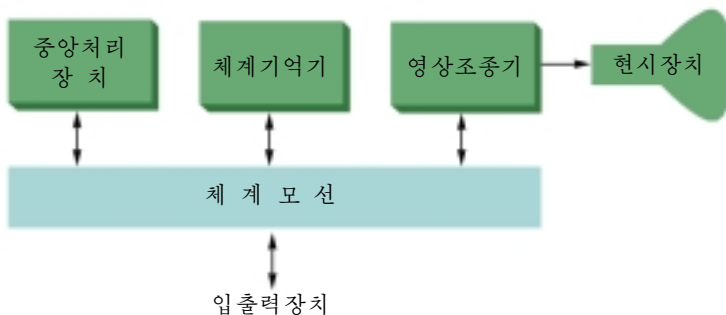



그림 2-25. 간단한 라스터도형처리체계의 구성

영상조종기

그림 2-26 에서는 라스터체계의 일반적인 구성을 보여 주었다. 체계기억기의 고정된 구역이 프레임완충기로 예약되며 영상조종기는 프레임완충기의 기억기를 직접 호출할수 있게 되여 있다.

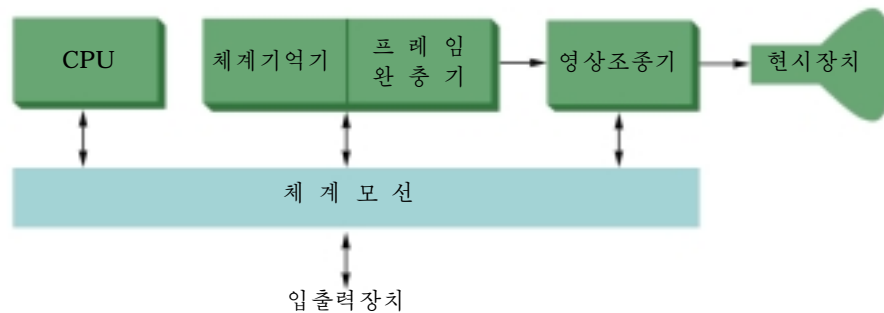



그림 2-26. 체계기억기의 고정된 부분이 프레임완충기로 예약된 라스터체계의 구성

프레임완충기위치, 그에 대응하는 화면에서의 위치는 직각자리표로 지적된다. 많은 도형현시장치들에서 자리표원점은 화면의 왼쪽 아래구석으로 잡는다(그림 2-27). 그러면 화면은 2차원자리표계의 첫 4분구와 같이 표현되며 정의 x값은 오른쪽으로 증가하고 정의 y값은 밑에서 우로 증가한다(일부 개인용콤퓨터에서는 자리표원점을 화면의 왼쪽 웃구석으로 잡는다. 그러므로 y값이 반전된다.). 이때 주사선에는 화면의 꼭대기 y_{max}로부터 아래 0까지 표식을 붙인다. 매개 주사선을 따라 화면의 화소위치는 0부터 x_{max}까지로 표식된다.

그림 2-28에 영상조종기의 기본재생동작흐름이 표시되였다. 화면화소의 자리표를 기억시키는데 두개의 등록기가 리용되고 있다. 초기에 x등록기는 0으로 설정되고 y등록기는 y_{max}로 설정된다. 다음에 프레임완충기에 기억시킨 이 화소위치에 대한 값을 꺼내여 CRT속의 세기를 설정하는데 리용한다. 그다음에 x등록기는 1만큼 증가되여 꼭대기주사선에서 다음화소에 대한 처리가 진행된다. 이 절차는 주사선의 매개 화소에 대하여 반복된다. 꼭대기주사선의 마지막화소가 처리된후 x등록기는 0으로 재설정되고 y등록기는 1만큼 감소된다. 다음 이 주사선의 화소들이 차례로 처리되며 절차는 매 련속된 주사선에 대하여 반복된다. 바닥주사선(y=0)의 모든 화소들까지 처리한후 영상조종기는 꼭대기주사선의 첫 화소위치에로 등록기들을 재설정하며 재생처리가 또다시 시작된다.

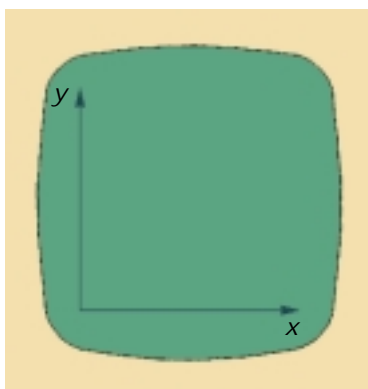



그림 2-27. 화면위치를 표현하기 위한 자리표계의 원점은 보통 왼쪽 아래구석으로 잡는다.

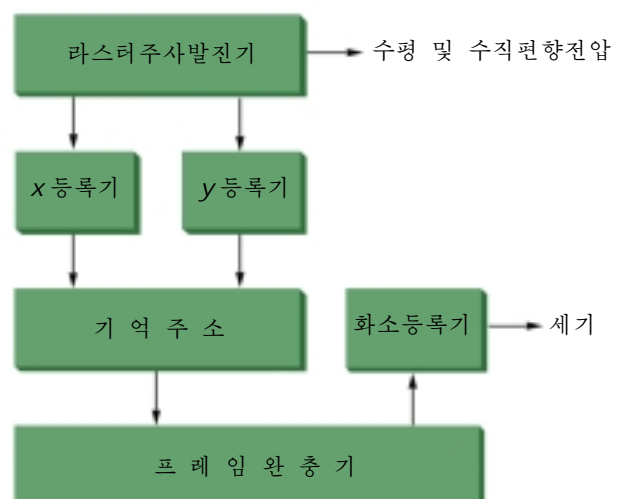



그림 2-28. 영상조종기의 기본재생동작

화면은 초당 60프레임의 속도로 재생되여야 하므로 그림 2-28에서 설명되는 절차를 일반적인 RAM소편에 의하여서는 처리할수 없다. 순환속도가 너무 느리다. 화소의 처리속도를 높이기 위하여 영상조종기는 매개 통과에서 재생완충기로부터 여러 화소값을 꺼낼수 있다. 여러 화소의 세기들은 그 다음에 개별적인 등록기들에 기억되여 린접한 화소들의 모임에 대한 속세기를 조종하는데 리용된다. 이 화소모임이 처리되였을 때 화소값의 다음 블로크가 프레임완충기로부터 꺼내 진다.

영상조종기는 기본재생조작외에도 많은 조작들을 수행한다. 여러가지 목적에 응용하기 위하여 영상조종기는 서로 다른 기억구역에서 서로 다른 재생주기에 화소세기값을 꺼낼수 있다. 실례로 고급한 체계들에서 흔히 두개의 프레임완충기를 리용하여 하나의 완충기가 세기값으로 채워 지는 동안 다른것은 재생에 리용할수 있게 한다. 그러면 두 완충기는 역할을 절환하게 된다. 이것은 움직이는 물체의 서로 다른 상을 련속적으로 재생완충기에 적재할수 있게 하므로 실시간동화생성을 위한 빠른 동작기구를 제공한다. 또한 영상조종기에 의하여 일부 변환들도 수행할수 있다. 화면구역들을 재생할 때 확대축소 또는 한 곳에서 다른 곳으로 움직일수 있다. 이와 함께 영상조종기는 흔히 검색표를 가지고 있는데 이렇게 하면 프레임완충기안의 화소값들이 CRT속세기를 직접 조종하는것이 아니라 검색표를 호출하는데 리용된다. 이것은 화면의 세기값을 빨리 변화시킬수 있는 방법을 준다. 검색표는 4장에서 보다 자세히 설명한다. 마지막으로 일부 체계들에서는 영상조종기가 프레임완충기화상을 텔레비죤카메라 또는 다른 입력장치로부터의 입력화상과 결합할수도 있게 설계되여 있다.

라스터주사현시처리기

그림 2-29에서는 때때로 **도형처리조종기**(graphics controller) 또는 **현시협동처리기**(display coprocessor)라고도 부르는 개별적인 **현시처리기**(display processor)로 이루어 진 라스터체계의 한가지 구성안을 보여 주었다. 현시처리기의 목적은 CPU를 도형처리의 자질구레한 일에서 해방시키는것이다. 또한 체계기억기밖에 현시처리기의 개별적인 기억구역을 제공할수도 있다.

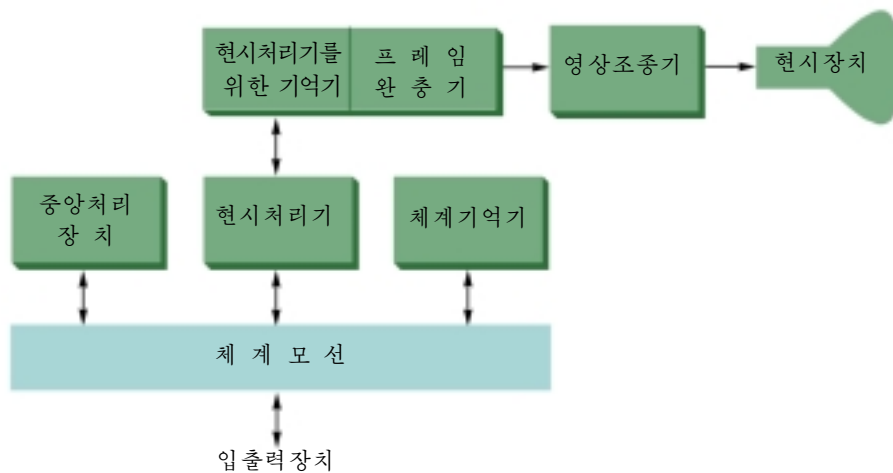



그림 2-29. 현시처리기를 가지는 라스터도형처리체계의 구성

현시처리기의 중요한 과제는 응용프로그람에서 주어 지는 그림의 정의를 프레임완충기안에 기억시키기 위한 화소세기값모임으로 수자화하는것이다. 이 수자화처리를 **주사변환**(scan conversion)이라고 한다. 직선과 기타 기하학적물체들을 지적하는 도형처리명령들은 불련속세기점들의 모임으로 주사변환된다. 실례로 선분을 주사변환한다는것은 직선경로에 제일 가까운 화소위치를 알아 내고 그 매개 위치에 대한 세기를 프레임완충기에 기억시킨다는것을 의미한다. 곡선 및 다각형의 륜곽선을 주사변환하는데도 류사한 방법이 리용된다. 문자는 그림 2-30에서와 같이 직4각형격자를 가지고 정의할수도

있고 또 그림 2-31에서와 같이 륜곽곡선으로 정의할수도 있다. 문자격자의 배렬의 크기는 5×7로부터 9×12 또는 고급한 현시장치들에서 그이상일수 있다. 문자격자는 직4각형격자무늬를 프레임완충기안의 지적된 자리표위치에 덧놓는것에 의해 현시된다. 륜곽곡선으로 정의되는 문자에서는 문자의 형태가 프레임완충기에 주사변환된다.

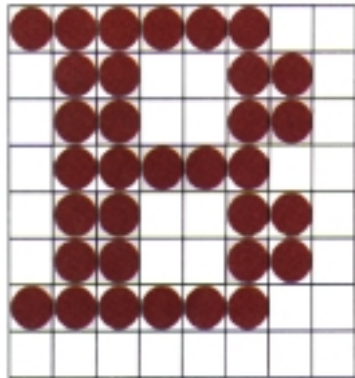



그림 2-30. 화소위치의 직4각형 격자로 정의되는 문자

그림 2-31. 륜곽곡선으로 정의되는 문자

현시처리기는 또한 다른 동작들도 수행하도록 설계된다. 이런 기능들에는 여러가지 형태의 선(파선, 점선, 실선)그리기, 색구역현시, 현시되는 물체에 대한 어떤 변환과 처리의 수행 등이 속한다. 또한 현시처리기는 대체로 마우스와 같은 대화식입력장치와 결합될수 있도록 설계된다.

라스터체계에서 기억기에 대한 요구를 줄이기 위하여 프레임완충기를 련결목록과 같이 구성하고 세기정보를 부호화하는 방법들이 나왔다. 이렇게 하는 한가지 방법으로서는 매개 주사선을 옹근수쌍의 모임으로 기억시키는것이다. 매개 쌍에서 첫번째 수는 세기값을 가리키며 두번째 수는 주사선에서 그 세기값을 가지는 서로 린접된 화소의 개수를 지적한다. **행정길이부호화**(run-length encoding)라고 하는 이 기술은 만약 그림이 대부분 각각 한가지 색의 긴 행정으로 되여 있는 경우 기억공간을 상당히 절약할수 있다. 류사한 방법은 화소세기가 선형적으로 변할 때도 취할수 있다. 다른 한가지 방법은 라스터를 직4각형구역들의 모임으로 부호화하는것이다(**세포부호화**(cell encoding)). 행정부호화의 결함은 세기변화를 나타내기 힘들고 실제로는 기억기용량이 행정의 길이가 감소할 때 증가하는것이다. 더우기 짧은 행정이 많이 포함되였을 때에는 현시처리기가 라스터를 처리하기 힘들다.

3절. 우연주사체계

그림 2-32에 간단한 우연주사(벡토르)체계의 구성을 보여 주었다. 응용프로그람은 도형처리프로그람과 함께 체계기억기에 입력되고 기억된다. 응용프로그람에 있는 도형처리명령들은 도형처리프로그람들에 의하여 현시파일로 번역되여 체계기억기에 기억된다. 그다음에 현시처리기는 화면을 재생하기 위하여 이 현시파일을 호출한다. 현시처리기는 현시파일에 있는 모든 명령을 매 재생주기에 한번씩 반복순환한다. 때때로 우연주사체계에서의 현시처리기를 현시처리단 또는 도형처리조종기라고도 한다.

우연주사체계에서 도형무늬는 전자속을 그림의 요소선들을 따라 편향시켜 그린다. 선은 그의 자리표끝점들에 대한 값에 의하여 정의되며 이 입력자리표값들은 X 및 Y편향전압으로 변환된다. 그러면 주어 진 끝점들사이의 구간을 채울수 있게 속이 편향되여 한번에 한개 선씩 그린다.

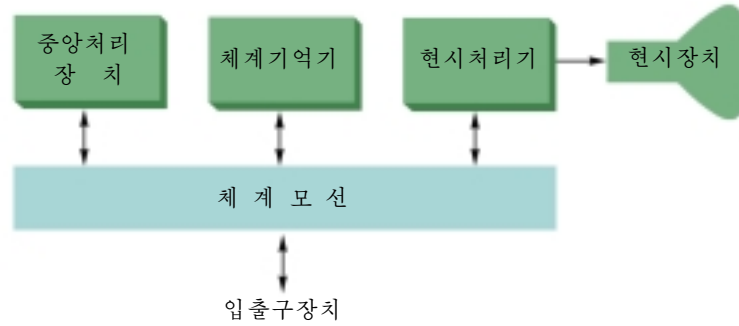



그림 2-32. 간단한 우연주사체계의 구성

4절. 도형현시장치 및 워크스테이션

현재 대부분의 도형현시장치들은 라스터주사현시장치로 동작한다. 여기서는 흔히 볼수 있는 도형처리하드웨어구성의 일부를 개괄한다. 도형처리체계에는 도형처리를 진행할수 있는 일반목적극소형콤퓨터체계(그림 2-33)로부터 도형처리응용을 위하여 전용으로 설계되는 고급한 완전색체계(그림 2-34)까지 있다. 화면의 분해능과 체계의 기타 능력들은 체계의 크기와 가격에 따라 다르지만 그림 2-33에 보여 준 Apple Quadra와 같은 개인용콤퓨터체계에서는 일반적으로 화면분해능이 640×480이다. 일반목적개인용콤퓨터체계에서 화면의 대각선치수는 12×21in범위에 있으며 선택가능한 색은 16～32000범위에 있다. 그림 2-34에 보여 준 체계와 같이 전문도형처리응용을 위하여 설계된 워크스테이션에서는 대표적인 화면분해능이 1280×1024이고 화면대각선은 16in 또는 그이상이다. 도형처리워크스테이션은 화소당 8～24bit(완전색체계), 높은 화면분해능, 고속처리기, 고성능체계에서 사용할수 있는 기타 선택항목들로 구성된다.

그림 2-33. 도형처리응용에 리용될수 있는 탁상용일반목적콤퓨터체계

그림 2-34. 건반과 마우스입력장치를 가진 콤퓨터도형처리워크스테이션

그림 2-35에서는 항공운수조종, 모의, 의학화상그리기 그리고 CAD와 같은 응용분야에 리용되는 고해상도도형현시장치를 보여 주었다. 이 체계에서 화면의 대각선길이는 27in이고 분해능은 2048×1536～2560×2048이며 재생속도는 비간격식주사로 80Hz 또는 60Hz이다.

그림 2-36에 보여 준 MediaWall이라고 하는 다중화면체계로는《벽크기》만한 큰 현시구역을 얻을 수 있다. 이 체계는 무역전람회, 협의회, 백화점, 박물관, 려객역에서와 같은 밝은 조명환경에서 큰 화면구역의 현시를 요구하는데 리용하기 위하여 설계되였다. MediaWall은 화상을 몇개 부분으로 가르고 매 구역을 도형정합기와 위성조종장치를 리용하여 현시장치 또는 투영기배렬에 분배하여 동작시킨다. 분해능이 640×480인 5×5 현시장치의 배렬일 때 MediaWall에서 정적장면 및 동화에 대한 전체적인 분해능은 3200×2400이다. 화면은 그림 2-36에서와 같이 격자에 지지하여 현시될수도 있고 또는 여러 부분들사이에 끊어 짐이 없이 이어 진 그림을 현시하기 위하여 격자를 없앨수도 있다.

그림 2-35. 초고분해능(2560 × 2048)의 색현시장치

그림 2-36. MediaWall 다중화면현시체계(3 × 3 의 현시장치배렬에 현시된 화상)

그림 2-37에 보여 준바와 같이 대부분의 도형처리워크스테이션들은 두개의 현시장치로 구성된다. 하나의 현시장치는 물체 또는 장면의 모든 특징을 보여 주고 두번째 현시장치는 그림의 일부를 자세히 현시하는데 리용된다. 혹은 2중현시장치체계에서 한 현시장치에서는 그림을 보이고 다른 현시장치는 그림의 요소들을 처리하기 위한 도형처리선택항목(차림표)들을 현시하는데 리용한다.

그림 2-37. 단일 및 2중현시장치의 도형처리워크스테이션

그림 2-38과 2-39에서는 여러가지 입력 및 기타 장치를 가지고 있는 대화식도형처리워크스테이션의 실례를 보여 주었다. 그림 2-38에서는 CAD응용에서의 대표적인 실례를 보여 주었다. 여러가지 건반, 누름단추통, 도형입력판, 마우스들이 설계공정에서 리용되도록 영상현시장치에 접속되여 있다. 그림 2-39에서는 미술가용워크스테이션의 일부 형태를 보여 주었다.

그림 2-38. CAD집단을 위한 다중워크스테이션

그림 2-39. 색라스터현시장치, 건반, 손유표가 달린 도형입력판, 빛판 그밖에 자료기억기와 원격통신장치가 있는 미술가용 워크스테이션

5절. 입력장치

도형처리워크스테이션에서 자료를 입력하기 위하여 여러가지 장치들을 사용할수 있다. 대부분의 체계들은 건반과 함께 자료를 대화적으로 입력시키기 위하여 특별히 설계된 하나 또는 그이상의 보충적인 장치들을 가지고 있다. 이런 장치들에는 마우스, 추적볼, 공간볼, 조종간, 수자화기, 다이얄, 누름단추통들이 속한다. 특수한 응용에서 리용되는 몇가지 다른 입력장치들로서는 자료장갑, 다침판, 화상스캐너, 음성체계가 있다.

건반

도형처리체계에서 문자수자건반은 기본적으로 문자렬을 넣기 위한 장치로 리용된다. 건반은 도형현시와 관련되는 그림의 표식과 같은 비도형적인 자료를 입력하는 효과적인 장치이다. 건반은 또한 화면자리표, 차림표선택, 도형함수의 입력을 쉽게 하는 측면도 있다.

유표조종건과 기능건은 일반목적건반들에서 공통적으로 가지고 있다. 기능건은 사용자가 자주 리용하는 조작을 하나의 건누르기로 처넣게 하며 유표조종건은 화면유표의 위치를 지적하여 현시되는 물체 또는 자리표위치를 선택하는데 리용할수 있다. 일부 건반들에는 추적볼, 조종간과 같은 다른 형의 유표위치지정장치들도 붙어 있다. 흔히 건반에는 수자자료를 빨리 입력하기 위하여 수자판이 추가적으로 더 포함되여 있다. 일반목적건반의 대표적인 실례들을 그림 2-1, 2-33, 2-34에서 보여 주었다. 그림 2-40에서는 인간공학적인 건반설계를 보여 주었다.

그림 2-40. 밑판대기를 움직일수 있게 인간공학적으로 설계된 건반 (건반의 매개 절반의 경사는 각각 조종할수 있다.)

특수한 경우 도형처리응용을 위한 입력을 자료값이나 도형처리조작을 선택하는 누름단추, 다이얄, 스위치들로부터 얻을수 있다. 그림 2-41에 누름단추통과 입력다이얄모임의 실례를 주었다. 누름단추와 스위치는 흔히 미리 정의된 기능을 입력하는데 리용되며 다이얄은 크기값입력을 위한 일반장치이다. 정의된 범위안의 실수는 다이얄회전에 의하여 입력선택된다. 가변저항기는 다이얄회전을 측정하는데 리용되며 그다음에 유표이동을 위한 편향전압으로 변환된다.

ㄱ) ㄴ)

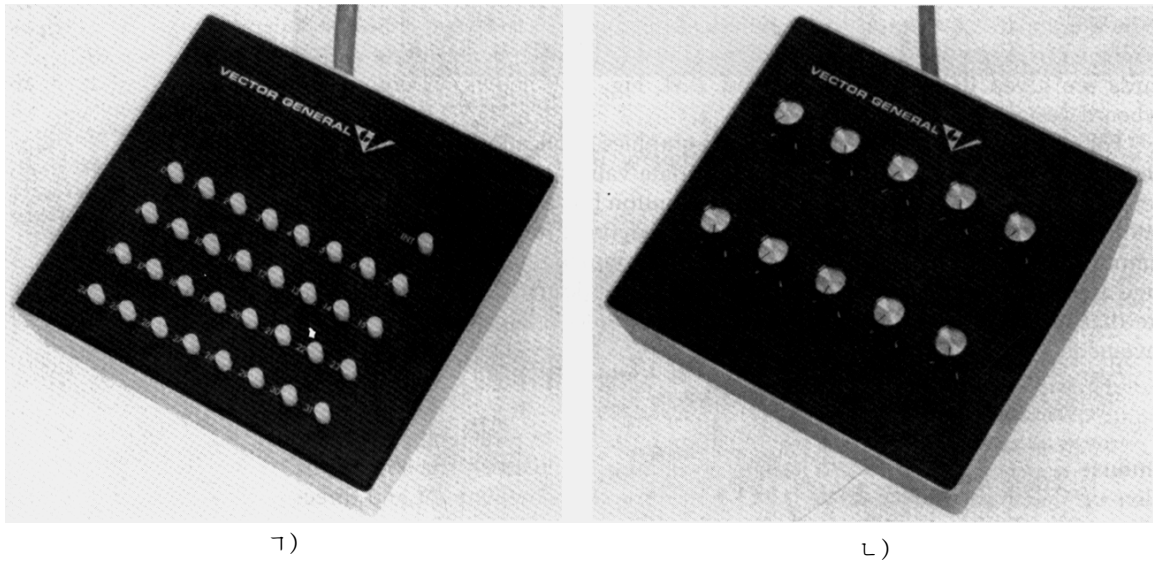



그림 2-41. 단추통(ㄱ)과 입력다이얄모임(ㄴ)

마우스

마우스는 화면유표의 위치를 지정하는데 리용되는 손바닥크기의 작은 통이다. 마우스의 밑에 있는 바퀴 또는 굴대는 이동의 크기와 방향을 등록하는데 리용된다. 마우스의 운동을 검출하는 다른 방법은 광학수감기에 의한것이다. 이 체계에서 마우스는 수평 및 수직선의 격자를 가지는 특별한 마우스판우에서 움직여 진다. 광학수감기는 격자의 선을 가로자르는 운동을 검출한다.

마우스는 유표를 이동시킴이 없이 들어서 다른 위치에 내려 놓을수 있으므로 화면유표위치의 상대적변화를 만드는데 리용된다. 유표위치의 등록이나 기능선택과 같은 일부 조작의 실행을 신호하기 위하여 보통 마우스의 꼭대기에는 하나, 둘 또는 세개의 단추가 있다. 지금 대부분의 일반목적도형처리체계들은 그림 2-1, 2-33, 2-34에서와 같이 마우스와 건반을 중요한 입력장치로 가지고 있다.

허용할수 있는 입력파라메터의 수를 증가시키기 위하여 기본적인 마우스구조에 추가적인 기구들이 포함될수 있다. 그림 2-42의 Z마우스는 세개의 단추 즉 측면에 엄지손가락바퀴, 꼭대기에 추적볼, 밑에 표준적인 마우스볼을 가지고 있다.

그림 2-42. Z마우스는 세개의 단추 즉 밑에 마우스볼, 측면에 엄지손가락바퀴, 꼭대기에 추적볼이 있다.

이 구조는 공간적인 위치, 회전, 기타 파라메터들을 선택하는 6개의 자유도를 가진다. Z마우스로 물체를 들어 올리고 회전시키고 임의의 방향으로 이동시킬수 있으며 또 3차원장면을 통하여 보기위치와 방향을 조종할수 있다. Z마우스는 가상현실, CAD, 동화 등에 많이 쓰인다.

추적볼 및 공간볼

추적볼(trackball)은 손가락 또는 손바닥으로 회전시킬수 있는 볼로서 그림 2-43 에서와 같이 화면에서 유표를 이동시키는데 쓴다. 볼에 접촉된 가변저항기는 회전량과 방향을 측정한다. 추적볼은 흔히 건반(그림 2-15) 또는 Z마우스(그림 2-42)와 같은 다른 장치들에 설치된다.

그림 2-43. 세 단추의 추적볼

추적볼은 2차원위치를 지정하는 장치이고 **공간볼**(spaceball)(그림 2-45)은 6개의 자유도를 주는 장치이다. 추적볼과 달리 공간볼은 실제로 움직이지는 않는다. 볼을 여러 방향에서 밀거나 당기면 변형측정기가 공간볼에 가해 진 압력을 측정하여 공간적인 위치와 방향을 결정한다. 공간볼은 가상현실체계, 모형화, 동화 기타 응용들에서 3 차원위치지정과 선택조작에 리용한다.

조종간

조종간(joystick)은 화면에서 유표를 사방으로 조종하는데 리용되는 받침대에 설치된 작은 수직지레대(손잡이라고 부른다)이다. 대부분의 조종간들은 화면위치를 손잡이의 실제적인 움직임으로 지적하지만 어떤것들은 손잡이에 미치는 압력에 응답한다. 그림 2-44에는 움직일수 있는 조종간을 보여 주었다. 일부 조종간은 건반에 설치되여 있기도 하지만 독립적으로 사용되는것도 많다.

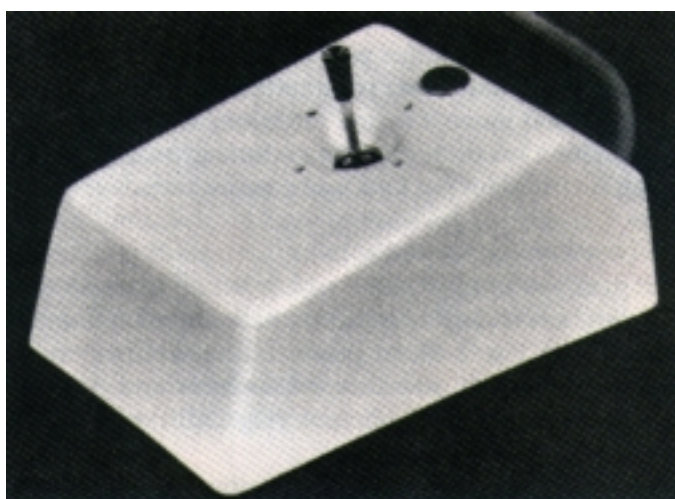



그림 2-44. 움직일수 있는 조종간

손잡이가 그의 중심위치로부터 임의의 방향으로 움직여 지는 거리는 화면의 해당한 방향에서의 유표이동에 대응한다. 조종간의 받침대에 설치된 가변저항기는 이 이동량을 측정하고 스프링은 손잡이를 풀어 놓을 때 그것을 중심위치로 돌려 보낸다. 하나 또는 그이상의 단추들은 화면위치가 선택되였을 때 일정한 동작을 신호하는 입력스위치와 같이 동작하도록 설정할수 있다.

다른 형의 움직일수 있는 조종간에서는 손잡이가 화면에서 유표를 일정한 속도로 지적된 방향으로 움직이게 하는 스위치를 동작시키는데 리용된다. 때때로 원을 따라 배렬된 8개의 스위치가 제공되게 되는데 이때 손잡이는 유표에 대하여 8개의 방향중 임의의 어느 하나를 선택할수 있게 한다. 압력

수감조종간(isometric 조종간이라고도 부른다.)에는 움직이지 않는 손잡이가 붙어 있다. 손잡이에 미치는 압력은 변형측정기로 측정되여 지정된 방향에로의 유표의 이동으로 변환된다.

자료장갑

그림 2-45에는 《가상》물체를 쥐는데 리용되는 **자료장갑**(data glove)을 보여 주었다. 장갑은 손과 손가락의 운동을 검출하는 여러가지 수감기로 만들어 졌다. 송신안테나와 수신안테나는 손의 위치와 방향에 대한 정보를 제공하는데 리용된다. 송수신안테나들은 각각 3차원의 직각자리표계를 형성하는 서로 수직인 3개의 선륜으로 구성된다. 장갑에서 입력되는 정보는 가상장면에서 물체의 위치를 규정하거나 그것을 처리하는데 리용되게 된다. 장면의 2차원투영은 영상현시장치에서 볼수 있으며 3차원투영은 모자를 통하여 볼수 있다.

그림 2-45. 자료장갑과 공간볼에서 입력된 자료로 2차원영상현시장치에 현시한 가상현실장면

수자화기

작도, 그림그리기, 물체우에서 자리표위치의 대화식선택을 진행하는 일반적인 장치는 **수자화기**(digitizer)이다. 이 장치는 2차원 및 3원공간에서 자리표값을 입력하는데 리용할수 있다. 일반적으로 수자화기는 도면 또는 물체를 주사하면서 곡선 또는 곡면형태를 근사선분들로 련결시킬수 있는 불련속자리표위치점들의 모임을 입력하는데 리용된다.

도형입력판(graphics tablet)(자료평판이라고도 한다.)은 수자화기의 한가지 형으로서 이것은 평면의 선택된 위치에서 손유표나 펜을 동작시켜 2차원자리표를 입력하는데 리용된다. 손유표는 위치를 조준하기 위한 십자쇠줄로 되여 있고 펜은 평판우에서 위치를 지적하는 연필모양의 장치이다. 그림 2-46과 2-47은 2, 4 혹은 16개의 단추와 손유표를 리용하는 탁상형 및 방바닥형도형입력판의 실례를 보여 주었다.

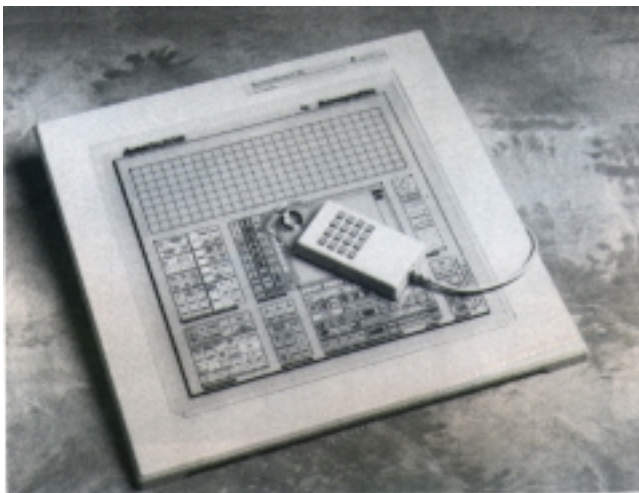



그림 2-46. 16단추의 손유표를 가지는 SummaSketch III 탁상형도형입력판

그림 2-47. 큰 그림의 수자화를 위하여 설계된 16단추의 손유표를 가지는 Microgrid III 도형입력판

도형입력판에서 펜입력의 실례를 그림 2-48과 2-49에 보여 주었다. 그림 2-49에서 보여 준 미술가의 수자화체계는 펜의 3차원위치를 검출하는데 전자기적공진을 리용하였다. 미술가는 도형입력판에서 서로 다른 압력으로 서로 다른 붓놀림을 할수 있다. 도형입력판의 크기는 탁상형 12×12in 로부터 방바닥형 44×60in 또는 그이상이다. 도형입력판은 탁상형에서 약 0.2mm, 더 큰것은 0.05mm 또는 그 이하의 높은 정밀도를 가지고 자리표위치를 선택할수 있다.

그림 2-48. 펜이 달린 탁상형 도형입력판

그림 2-49. 압력수감형코드 없는 펜이 있는 미술가의 수자화체계

대부분 도형입력판들은 평판면에 묻어 놓은 선들의 직4각형격자선들로 만들어 진다. 선을 따라 차례로 발생되는 전자기적임풀스신호는 도형입력판에서 위치를 등록하기 위한 활성화된 펜 또는 손유표에 있는 선륜에 유도된다. 도형입력판우의 위치를 결정하는데는 사용되는 기술에 따라 신호세기, 부호화된 임풀스, 위상차가 리용될수 있다.

음향(또는 음파)식도형입력판은 펜의 위치를 검출하는데 음파를 리용한다. 펜에서 전기적방전에 의하여 발생되는 소리는 선마이크 또한 점마이크를 리용하여 검출한다. 펜의 위치는 발생된 소리가 서로 다른 마이크위치에 도착하는 시간에 의하여 계산된다. 2차원음향식도형입력판의 우점은 마이크를 도형입력판작업구역을 형성하는 임의의 면에 놓을수 있는것이다. 이것은 책에 있는 그림을 수자화하는것과 같은 여러가지 응용에 편리하다.

3차원수자화기는 위치를 기록하는데 음 또는 전자기적전송을 리용한다. 전자기적전송방법은 자료장갑에서 리용되는것과 류사하다. 즉 송신기와 수신기를 리용하여 펜이 물체겉면우로 움직일 때 그의 위치를 계산한다. 그림 2-50은 Apple Macintosh콤퓨터에서 리용하기 위하여 설계된 3차원수자화기를 보여 주었다. 비금속물체우에서 점들이 선택될 때 면의 선그물구조륜곽선이 콤퓨터화면에 현시된다. 면의 륜곽선이 만들어 지면 물체의 현실적인 화면을 만들기 위하여 조명효과로 명암을 줄수 있다. 이 체계의 분해능은 모형에 따라 0.8~0.08mm이다.

그림 2-50. Apple Macintosh 콤퓨터에서 리용되는 3차원수자화체계

화상스캐너

그림 2-51. 본문 또는 도형을 입력하는데 리용되는 손바닥크기의 스캐너

도면, 그라프, 색 및 흑백사진, 본문은 **화상스캐너**(image scanner)를 리용하여 기억시킬 정보우로 광학식주사기구를 통과시킴으로써 콤퓨터가 처리를 할수 있게 기억시킬수 있다. 이때 흑백계조 또는 색갈의 점차적인 변화가 배렬에 등록 및 기억된다. 일단 그림의 내부표현을 얻게 되면 회전, 확대, 축소, 개별적인 화면구역에로 넘기는 변환 등을 적용할수 있다. 또한 여러가지 화상처리방법들을 적용하여 그림의 배렬표현을 수정할수 있다. 기억된 문서우에서 주사입력된 본문에 대한 여러가지 편집조작도 수행할수 있다. 일부 스캐너들은 도형 또는 본문을 주사할수 있으며 여러가지 크기와 능력을 가진다. 그림 2-51에는 손바닥크기의 작은 스캐너를 보여 주었고 그림 2-52와 2-53에서는 더 큰 형을 보여 주었다.

ㄱ)

ㄴ)

그림 2-52. 탁상형완전색스캐너: ㄱ- 600dpi 분해능을 가지는 평면형스캐너, ㄴ- 선택가능한 분해능이 50~4000dpi 인 원통형스캐너

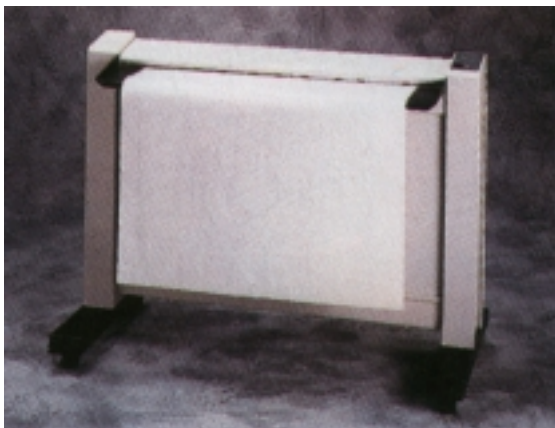



그림 2-53. 너비 40in, 길이 100ft 까지의 건축 및 공학도면을 주사하는데 리용되는 대형방바닥형스캐너

다침판

이름에 함축되여 있는바와 같이 **다침판**(touch panel)은 현시된 물체 또는 화면의 위치가 손가락의 접촉에 의하여 선택되게 한다. 다침판의 대표적인 응용은 도형적인 그림기호로 표현된 처리항목의 선택이다. 그림 2-54 에 보여 주는 플라즈마평판과 같은 일부 체계들은 다침화면으로 설계된다.

다른 체계들은 다침수감기구를 가지는 투명한 장치를 영상현시장치화면우에 덧놓는 방법으로 다침입력에 적응시키고 있다. 다침입력은 광학, 전기, 음향학적방법들을 리용하여 기록할수 있다.

ㄱ)

ㄴ)

그림 2-54. 다침화면을 가지는 플라즈마평판

광학식다침판은 틀의 수직변과 수평변을 따라 배렬된 적외선발광2극소자들을 쓴다. 반대쪽의 수직 및 수평변들에는 빛검출기가 있다. 이 검출기들은 판을 다칠 때 어느 빛속이 끊어 지는가를 기록하는데 리용된다. 끊어 지는 두개의 교차빛속은 선택된 화면위치의 수평 및 수직자리표를 나타낸다. 위치는 약 1/4 in의 정밀도로 선택될수 있다. 조밀하게 놓인 적외선발광2극소자에서는 두개의 수평속 또는 두개의 수직속이 동시에 끊어 질수 있다. 이 경우에는 두개의 끊어 진 속들사이의 평균위치가 기록된다. 적외선발광2극소자는 적외선주파수에서 동작하므로 사용자에게 빛이 보이지 않는다. 그림 2-55는 체계의 색과 륜곽에 맞출수 있게 설계된 광학식다침판에서의 적외선발광2극소자의 배렬을 보여 주었다.

그림 2-55. 광학식다침판, 틀의 변두리를 따라 적외선발광 2 극소자와 검출기가 배렬되여 있다.

전기식다침판은 좁은 간격으로 분리된 두개의 투명한 판으로 만든다. 한 판은 전도성재료를 칠하고 다른 판은 저항성재료를 칠한다. 바깥판을 다칠 때 내부판과 접촉하게 된다. 이 접촉은 저항판에 인가되는 전압강하를 만들어 선택된 화면위치의 자리표값으로 변환된다.

음향식다침판에서는 높은 주파수의 소리파(초음파)가 유리판을 따라 수평 및 수직방향으로 발생된다. 화면을 다치면 각 파의 일부분이 손가락으로부터 송파기에로 반사되게 된다. 접촉점의 화면위치는 각 파의 송신과 반사사이의 시간간격을 측정하여 계산한다.

빛펜

그림 2-56에 **빛펜**(light pen)의 한가지 형태의 모양을 보여 주었다. 이러한 연필모양의 장치는 CRT 화면의 점에서 나오는 빛을 검출하여 화면위치를 선택하는데 리용된다. 이 장치는 전자속이 매개 점을 때릴 때 형광면으로부터 방출되는 짧은 빛폭발에 민감하다. 방안의 배경빛과 같은 다른 광원은 일반적으로 빛펜에서 검출되지 않는다. 전자속에 의해 점이 빛을 낼 때 화면의 그 점을 가리키는 동작상태의 빛펜은 전자속의 자리표위치를 기록할수 있게 하는 전기적인 임풀스를 발생시킨다. 유표위치지정장치에서와 같이 기록되는 빛펜자리표는 물체의 위치를 지정하거나 처리항목을 선택하는데 리용할수 있다.

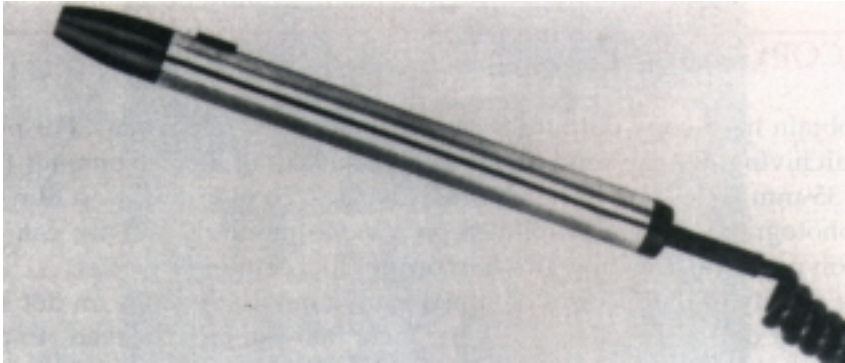



그림 2-56. 단추스위치에 의하여 동작상태로 되는 빛펜

비록 아직 빛펜이 남아있기는 하지만 그것은 이미 개발된 다른 입력장치들에 비해 볼 때 여러가지 결함을 가지고 있기때문에 그리 일반화되지 못하였다. 실례를 들면 빛펜으로 화면을 가리킬 때 화면화상의 일부가 손과 펜에 의하여 덮어 져 감추어 진다. 그리고 빛펜을 오래동안 쓰면 팔을 지치게 할수 있다. 또한 빛펜은 검은 구역안의 위치는 검출할수 없기때문에 일부 응용에서는 특별한 대책을 요구한다. 빛펜으로 화면구역안에서 모든 위치를 선택할수 있게 하자면 매개 화면화소에 령 아닌 세기를 주어야 한다. 더우기 빛펜은 때때로 방안의 배경조명때문에 거짓읽기를 줄수 있다.

음성체계

음성인식기는 일부 도형처리워크스테이션에서 음성지령을 받아 들이는 입력장치로 리용된다. **음성체계**(voice systems)에서의 입력은 도형처리조작을 시작하거나 자료를 넣는데 리용된다. 이런 체계들은 입력을 이미 정의된 단어와 줄인말사전과 대조한다.

그림 2-57. 음성인식체계

체계에서 쓰는 지령단어들을 말하는 개별적인 조작자에 대하여서는 사전을 만들어 둔다. 매개 단어를 여러번 말하면 체계는 단어를 분석하고 사전에 그 단어에 대한 주파수모양을 수행해야 할 기능과 함께 확정한다. 후에 음성명령이 주어 지면 체계는 사전의 주파수모양과 대조한다. 입력되는 음성은 일반적으로 그림 2-57에서와 같이 수화기가 달린 마이크에 들어 간다. 마이크는 다른 배경소리가 들어 가는것을 최소화하도록 설계된다. 다른 조작

자가 체계를 리용하려고 하면 사전을 그 조작자의 음성모양으로 재설정하여야 한다. 음성체계는 조작자가 지령을 넣을 때 한 장치에서 다른 장치로 주의를 돌리지 않아도 되므로 다른 입력장치들에 비해 일부 우점을 가진다.

6절. 경복사장치

화상의 경복사의 출력은 여러가지 형식으로 얻을수 있다. 화상파일을 보여 주거나 보관하기 위하여 35mm 환등필림 또는 투영필림을 만드는 장치나 봉사단위에 보낼수 있다. 화상을 필림에 옮기려면 간단히 영상현시장치에 현시되는 장면을 사진 찍을수 있다. 또 도형출력을 인쇄기나 작도기에 보내여 그림을 종이에 옮길수도 있다.

장치로부터 얻어 지는 화질은 점의 크기와 현시될수 있는 인치당 점의 개수 또는 인치당 선의 개수에 관계된다. 고급한 인쇄기들은 인쇄되는 본문문자렬에서 원활한 문자를 만들기 위하여 린접하는 점들이 겹치도록 점위치를 밀기한다.

인쇄기들은 충격 또는 비충격방법으로 출력을 만든다. 충격식인쇄기는 이미 만들어 진 문자의 생김새를 잉크띠에 눌러 반대쪽 종이에 찍는다. 행인쇄기는 충격식장치의 실례로서 피대, 사슬, 원통 또는 바퀴에 설치된 활자들을 가지고 있다. 비충격식인쇄기와 작도기는 종이에 화상을 얻을 때 레이자기술, 잉크분사분무기, 건식인쇄처리(사진복사기계에서 리용되는것과 같은), 정전기적방법, 전열방법 등을 쓴다.

충격식문자인쇄기는 흔히 가는 타자침들의 직4각형배렬이 있는 점행렬인쇄머리를 가지며 침의 개수는 인쇄기의 질에 관계된다. 개별적인 문자 또는 도형무늬는 인쇄될 무늬를 형성하도록 하기 위해 해당한 침들만 앞으로 나가고 나머지 침들은 나가지 않게 하는 방법으로 얻는다. 그림 2-58에서는 점행렬인쇄기에서 인쇄된 그림을 보여 주었다.

레이자장치에서는 레이자속으로 셀렌과 같은 광전재료를 칠한 회전원통에 전하분포상을 만든다. 잉크분말이 원통에 가해 지고 그다음 종이에 옮겨 진다. 그림 2-59 에는 분해능이 360dpi인 탁상레이자인쇄기의 실례를 보여 주었다.

잉크분사방법은 원통을 둘러 감싼 종이우에서 수평으로 움직이면서 잉크를 분출하는 방법으로 출력를 만든다. 전기적으로 대전된 잉크흐름은 점행렬무늬를 만들도록 전기마당에 의해 편향된다. 그림 2-60에서는 분해능이 360dpi인 탁상잉크분사작도기를 보여 주며 그림 2-61에서는 대형고분해능잉크분사인쇄기/작도기를 보여 주었다.

그림 2-58. 밝고 어두운 구역을 만들자면 점무늬의 밀도가 어떻게 변화되여야 하는가를 보여 주는 점행렬인쇄기에서 만들어 진 그림

그림 2-59. 소형평판식레이자인쇄기

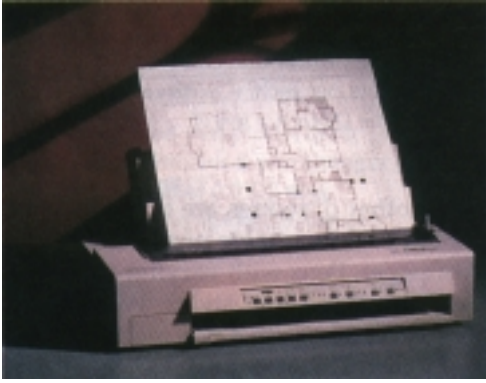



그림 2-60. 360dpi 탁상잉크분사작도기

ㄱ)

ㄴ)

그림 2-61. 1500 ~ 1800dpi 의 분해능을 얻기 위하여 점크기를 변화시키는 방바닥형의 잉크분사식인쇄기

정전기식장치는 부의 전하를 종이의 길이를 따라 한번에 한행씩 놓는다. 그다음에 종이에 잉크분말을 뿌린다. 잉크분말은 정으로 충전되여 있어 부로 충전된 구역에 끌리워 밀착되여 해당한 출력을 만든다. 그림 2-62 에는 색정전기식인쇄기/작도기를 보여 주었다. 전열방법에서는 점행렬인쇄머리에서 발생시킨 열로 무늬를 열수감종이에 출력한다.

충격식인쇄기에서는 서로 다른 색의 잉크띠를 리용함으로써 제한된 색출력을 얻을수 있다. 비충격식장치들에서는 색무늬를 만들기 위하여 3원색(푸른풀색, 분홍색, 누런색)을 결합하는 여러가지 기술을 리용한다. 레이자 및 건식인쇄장치들은 3개의 색소를 한번에 한가지 색씩 개별적으로 뿌리지만 잉크분사방법에서는 종이에 3개의 색을 매개 인쇄선을 따라 단번에 동시에 쏜다.

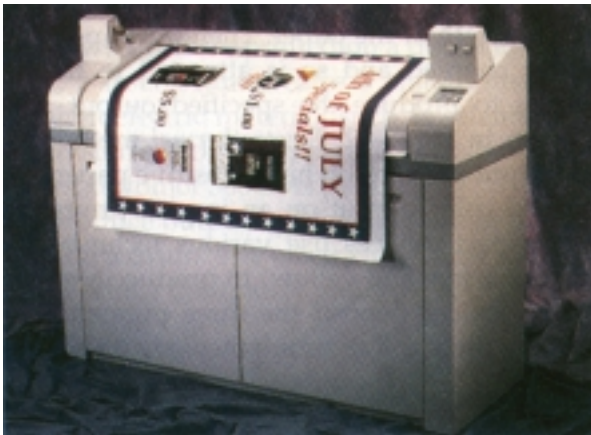



그림 2-62. 400dpi 로 현시할수 있는 정전기식인쇄기

설계도와 기타 도면은 일반적으로 잉크분사식 또는 펜작도기로 만든다. 펜작도기는 펜물개 또는 종이판을 가로지르는 가름대우에 설치된 하나 또는 그이상의 펜을 가진다. 서로 다른 색과 너비를 가지는 펜들이 여러가지 명암과 선형태를 만드는데 리용된다. 잉크펜, 볼펜, 펠트펜들은 모두 펜작도기에서 사용할수 있는것들이다. 작도종이는 팽팽하게 펴놓거나 원통이나 벨트에 감을수 있다. 펜이 가름대를 따라 앞뒤로 움직일 때 가름대는 움직일수도 있고 부동일수도 있다. 종이는 집게나 진공 또는 정전기적전하로 고정시킨다. 그림 2-63에서는 탁상평판식펜작도기의 실례를 주며 그림 2-64에서는 대형원통식펜작도기를 보여 주었다.

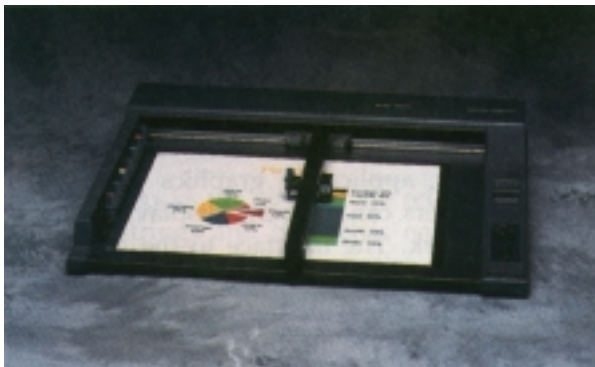



그림 2-63. 분해능이 0.025mm 인 탁상펜작도기

7절. 도형처리쏘프트웨어

도형처리쏘프트웨어에는 크게 나누어 두가지 즉 일반프로그람작성패키지와 전문목적응용패키지가 있다. 일반도형처리프로그람작성패키지는 C, FORTRAN과 같은 고수준프로그람작성언어에서 리용될수 있는 여러가지 도형처리함수들의 모임을 준다. 일반도형처리프로그람작성패키지의 실례로는 실리콘그라픽스회사의 GL (Graphics Library)체계이다. 일반패키지의 기초적인 함수들에는 그림요소(직선, 다각형, 원 및 기타 도형)의 발생, 색 및 세기값설정, 보기선택, 변환의 적용에 대한것들이 들어 있다. 반대로 응용도형처리패키지는 비프로그람작성자를 위하여 설계되는데 이때 사용자는 도형처리조작이 어떻게 진행되는가는 생각지 않아도 화면을 발생시킬수 있다. 이런 패키지에서 도형처리루틴들의 대면부는 사용자가 독자적으로 프로그람과 통신할수 있게 해준다. 이러한 응용패키지의 실례로는 미술가의 그림그리기프로그람과 여러가지 사무와 의학을 위한 체계들이다.

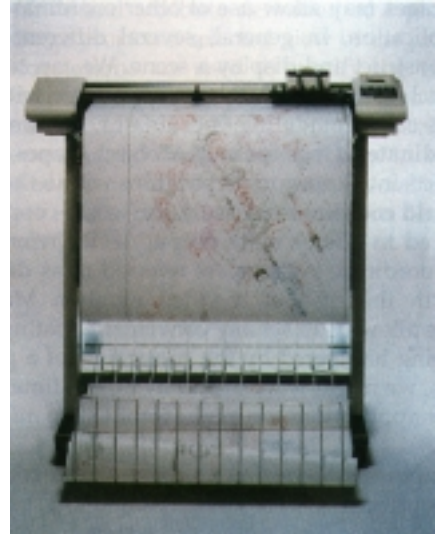



그림 2-64. 8가지의 색펜을 자동적으로 교체할수 있는 0.0127mm 분해능의 대형원통식펜작도기

자리표표현

일부 례외로 되는 경우도 있지만 일반도형처리프로그람들은 직각자리표서술을 쓰도록 설계되여있다. 만약 그림의 자리표값들이 일부 다른 자리표계(구면, 원기둥자리표)로 주어 지면 그것은 도형처리프로그람에 입력되기전에 직각자리표로 변환되여야 한다. 전문목적의 프로그람들은 응용에 맞는 다른 자리표계를 리용할수도 있다. 일반적으로 장면을 만들고 현시하는데는 여러가지 서로 다른 직각자리표계가 리용된다. 나무나 가구와 같은 장면안의 개별적인 물체들의 형태는 **모형화자리표**(modeling coordinates) 또는 때때로 **국부자리표**(local coordinates), **주자리표**(master coordinates)라고 하는 독립적인 자리표계안에서 만들어 진다. 개별적인 물체들의 형태가 주어 지면 물체들을 장면안의 적당한 위치에 **세계자리표**(world coordinates)라고 하는 자리표계를 리용하여 놓게 된다. 마지막으로 장면의 세계자리표서술은 하나 또는 그이상의 출력장치의 자리표로 변환되여 현시된다. 이 현시자리표계를 **장치자리표**(device coordinates)**계** 또는 영상현시장치인 경우 **화면자리표**(screen coordinates) **계**라고 한다. 모형화 및 세계자리표의 정의는 매개 출력장치의 구속을 받음이 없이 임의의 편리한 류점수 또는 옹근수값을 설정할수 있게 한다. 물체의 치수에 대하여 말하면 어떤 장면들에서는 몇분의 1ft 로 지적하고 싶을수 있고 한편 다른 응용에서는 미리메터, 키로메터, 광년을 리용하고 싶을수도 있다.

일반적으로 도형처리체계는 마지막에 세계자리표위치를 구체적인 장치자리표로 변환하기전에 0~1사이에서 **정규화된 장치자리표**(normalized device coordinates)로 먼저 변환된다. 이것은 체계로 하여금 개개의 워크스테이션에서 리용될수 있는 여러가지 장치에 의존하지 않게 한다. 그림 2-65에서는 2차원응용에서 모형화자리표로부터 장치자리표에로의 자리표변환순서를 보여 주었다. 이 설명에서 초기의 모형화자리표위치(x_{mc}, y_{mc})는 다음의 순서로 장치자리표위치(x_{dc}, y_{dc})로 변환된다.

$$(x_{mc}, y_{mc}) \to (x_{wc}, y_{wc}) \to (x_{nc}, y_{nc}) \to (x_{dc}, y_{dc})$$

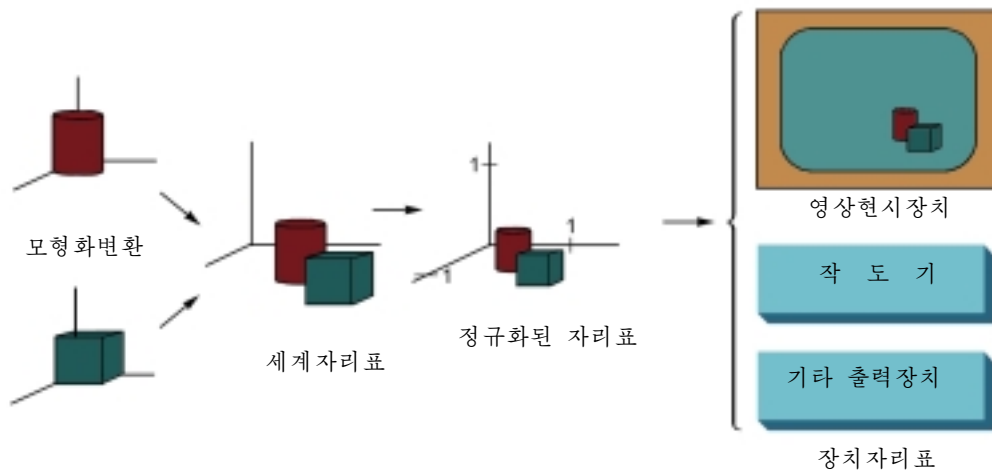



그림 2-65. 2차원장면에서 모형화자리표로부터 장치자리표에로의 변환순서(물체의 형태는 국부적인 모형화자리표계에서 정의되고 그다음에 세계자리표의 전체 장면안에서 위치가 정해 진다. 세계자리표서술은 그다음에 정규화된 자리표로 변환된다. 마지막단계에서 개별적인 장치구동기들이 장면의 정규화된 자리표표현을 현시용출력장치에 옮긴다.)

이 변환에서 모형화 및 세계자리표위치는 임의의 류점수값으로 될수 있으며 정규화된 자리표는 부등식 $0 \le x_{nc} \le 1$, $0 \le y_{nc} \le 1$을 만족시키며 장치자리표 x_{dc}와 y_{dc}는 매개 출력장치에서 (0,0)부터 (x_{max}, y_{max})사이의 옹근수이다. 척도와 종횡비에서의 차이를 고려하면서 정규화된 자리표는 적당한 비례가 유지되도록 출력장치의 바른4각형구역에 넘어 간다.

도형처리함수

일반목적도형처리프로그람은 사용자에게 그림을 만들고 처리하는 여러가지 함수들을 제공해 준다. 이 루틴들은 그것들이 출력, 입력, 속성, 변환, 보기, 일반조종을 다루는가 하는데 따라 분류할수 있다.

그림의 기초적인 건축블로크를 **출력기초요소**(output primitives)라고 한다. 여기에는 문자렬과 점, 직선, 곡선, 채운구역(다각형, 원 등)과 같은 기하학적실체 그리고 색점의 배렬로 정의되는 형체들이 속한다. 출력기초요소발생루틴은 그림을 만드는 가장 기초적인 도구를 준다.

속성(attributes)은 출력기초요소들의 특성이다. 즉 속성은 매개 기초요소들이 어떻게 현시되는가를 서술한다. 여기에는 세기와 색, 선형태, 본문형태, 구역채우기무늬가 속한다. 이 부류의 함수들은 개별적인 기초요소 또는 출력기초요소들의 집단에 대하여 속성을 설정하는데 리용될수 있다.

장면안에서 물체의 크기, 위치, 방향은 **기하학적변환**(geometric transformations)을 리용하여 변화시킬수 있다. 이와 류사하게 **모형화변환**(modeling transformations)은 모형화자리표로 주어 지는 물체서술을 리용하여 장면을 만드는데 리용한다.

그림의 기초요소들과 속성의 정의가 주어 지면 도형처리프로그람은 해당한 그림의 보임상을 출력장치에 투영한다. **보기변환**(viewing transformations)은 보여 주어야 할 보임상과 리용될 출력현시구역의 부분을 지적하는데 리용된다. 그림은 사용하는 쏘프트웨어에 따라 **구조물**(structures) 혹은 **토막**(segment) 혹은 **물체**(object)라고 부르는 요소부분들로 다시 분할할수 있다. 매개 구조물은 그림의 하나의 론리적단위를 정의한다. 여러개의 물체를 가지는 장면은 매개 물체를 개별적으로 이름을 가진 구

조물로 참조할수 있다. 구조물처리루틴들은 구조물의 생성, 수정, 변환과 같은 조작을 수행한다.

대화식도형처리응용프로그람들은 마우스, 도형입력판, 조종간과 같은 여러가지 종류의 입력장치를 리용한다. 입력함수들은 이 대화식장치들로부터의 자료흐름을 조종 및 처리하는데 리용된다.

마지막으로 도형처리프로그람은 현시화면의 지우기와 파라메터의 초기화와 같은 몇가지 보조적인 일감처리함수들을 가지고 있다. 이런 자질구레한 일들을 수행하는 함수를 조종조작이라는 이름으로 총괄할수 있다.

쏘프트웨어표준

표준화된 도형처리쏘프트웨어의 근본목적은 이식성이다. 프로그람을 표준도형처리함수들로 설계하면 쏘프트웨어를 한 하드웨어체계로부터 다른데로 옮기기 쉽고 서로 다른 실현 및 응용에 리용할수 있다. 표준이 없이 어느 한 하드웨어체계에 대하여 설계된 프로그람은 흔히 프로그람의 방대한 재쓰기없이는 다른 체계에로 옮기기 곤난하다.

많은 나라의 국제 및 민족표준화계획기구들이 콤퓨터도형처리에서 일반적으로 받아 들일수 있는 표준을 개발하는데서 협동하였다. 표준화를 위한 고심어린 노력을 걸쳐 **도형처리핵심체계 GKS**가 개발되였다. 이 체계는 **국제표준화기구 ISO**와 **미국규격협회 ANSI**를 포함한 여러 민족표준화기구들에 의하여 첫 도형처리쏘프트웨어표준으로 채용되였다. GKS는 원래 2차원도형처리프로그람으로 설계되였는데 그후에 3차원GKS확장판이 개발되였다. 표준화기구들에 의하여 개발되고 인정된 두번째 쏘프트웨어표준은 **PHIGS**이다. PHIGS는 GKS의 확장으로서 물체의 모형화, 색지적, 면실감처리 그리고 그림조작능력이 더 강하다. 후에 PHIGS+라고 하는 PHIGS의 확장판이 개발되여 PHIGS에서는 사용불가능했던 3차원면명암처리를 할수 있게 되였다.

표준도형처리함수들은 그 어떤 프로그람작성언어에도 의존하지 않는 설명모임으로 정의된다. 그 다음 매개 고수준프로그람작성언어에서 언어판이 정의된다. 이것은 해당 언어에서의 여러가지 표준도형처리함수의 호출에 대한 문법을 준다. 실례로 련결된 n-1개 2차원선분들의 순서지적에 대한 PHIGS(GKS)함수의 일반형식은

```
polyline (n, x, y)
```

이다. FORTRAN언어에서는 이 절차가 이름이 GPL인 보조루틴으로 실현되였다. FORTRAN을 리용하는 도형처리프로그람작성자는 이 절차를 보조루틴호출명령문 CALL GPL(N, X, Y)으로 호출할수 있다. 여기서 X 및 Y는 선의 끝점들에 대한 자리표값들의 1차원배렬이다. C에서는 절차를 polyline(n, pts)로 호출한다. 여기서 pts는 자리표끝점위치들의 목록이다. 매개 언어에서의 모양은 대응하는 언어의 능력을 제일 잘 리용하며 자료형, 파라메터넘기기, 오유와 같은 여러가지 문법적요구에 맞게 정의된다.

다음장들에서는 기초적인 도형처리개념들과 도형처리프로그람의 설계 및 응용을 설명하기 위한 체계로서 PHIGS에서 정의되는 표준함수들을 리용한다. 도형처리함수들의 실현을 위한 알고리듬들과 함수들의 몇가지 응용을 설명하기 위한 실례프로그람들은 C언어로 준다. PHIGS정의에 기초한 함수의 서술적인 이름은 도형처리함수가 프로그람에서 참조될 때에 리용된다.

PHIGS는 기초적인 도형처리함수들에 대한 설명은 주고 있으나 출력장치의 도형대면부를 위한 표준적인 방법론은 제공하지 않고 있다. 그리고 또 그림의 기억 및 전송에 대한 방법도 설명하지 않고 있다. 이 부분에 대하여서는 개별적인 표준들이 개발되였다. 장치대면방법에 대한 표준화는 **콤퓨터도형대면부 CGI**(Computer Graphics Interface)체계에서 주고 있다. 그리고 **콤퓨터도형처리메타파일 CGM**(Computer Graphics Metafile)체계는 그림의 파일보관 및 전송에 대한 표준을 지적하고 있다.

PHIGS의 워크스테이션

일반적으로 워크스테이션이라는 용어는 단일사용자를 위하여 설계된 여러가지 입출력장치들을 갖추고 있는 콤퓨터체계를 의미한다. 그러나 PHIGS와 CKS에서는 **워크스테이션**이라는 용어를 도형처리하드웨어와 쏘프트웨어의 여러가지 조합을 식별하는데 리용한다. PHIGS에서 워크스테이션은 하나의 출력장치, 하나의 입력장치, 입출력장치들의 결합, 파일 또는 지어 영상현시장치에 현시되는 창문으로 될수 있다.

응용프로그람안에서 여러가지 《워크스테이션》을 정의 및 리용하자면 워크스테이션식별자와 워크스테이션형을 지적하여야 한다. 다음의 명령문들은 PHIGS프로그람의 일반적인 구조를 준다.

```
openPhigs (errorFile, memorySize)
openWorkstation(ws, connection, type)
   { create and display picture }
closeWorkstation (ws)
closePhigs
```

여기서 파라메터 errorFile은 발생하는 모든 오유통보문을 넣어 두기 위한것이며 파라메터 memorySize는 내부기억구역의 크기를 지적한다. 워크스테이션식별자(옹근수)는 파라메터 ws로 주어 지고 파라메터 connection은 워크스테이션에 대한 호출방법을 말한다. 그리고 파라메터 type는 입력장치, 출력장치, 입출력장치의 결합, 입력 또는 출력메타파일과 같은 워크스테이션의 개별적인 종류를 지적한다.

여러개의 입력장치로부터 오는 입력과 여러개의 출력장치에로 향하는 출력을 가지는 개별적인 응용에서는 임의의 개수의 워크스테이션을 열어 놓을수 있다. 그림의 생성 및 처리의 기본절차를 고찰한후 8장에서 응용프로그람들에서의 입출력방법을 설명한다.

요약

이 장에서는 콤퓨터도형처리체계의 기본하드웨어와 쏘프트웨어의 면모를 훑어 보았다. 하드웨어의 구성요소에는 영상현시장치, 경복사장치, 건반,도형입출력을 위한 기타 장치들이 속한다. 도형처리쏘프트웨어에는 전문목적의 응용프로그람패키지와 일반프로그람작성패키지가 있다.

주되는 도형현시장치는 텔레비죤기술에 기초한 라스터재생현시장치이다. 라스터체계는 화면의 매 위치(화소)에 대한 세기정보를 기억시키기 위하여 프레임완충기를 사용한다. 그러면 CRT에서 전자속이 우에서부터 아래로 매 주사선을 따라 가면서 프레임완충기로부터 이 정보를 다시 꺼내여 화면우에 그림을 만든다. 좀 낡은 벡토르장치는 지적된 끝점들사이의 선들을 긋는 방법으로 그림을 만든다. 이 경우 그림정보는 선긋기명령그림모임으로 기억된다.

기타 많은 영상현시장치들을 사용할수 있다. 특히 평판현시장치기술이 빠른 속도로 개발되고 있으며 가까운 앞날에 이 장치들이 대부분 라스터현시장치들을 대신할수 있을것이다. 오늘날 평판현시장치는 일반적으로 작은 체계와 전문목적의 체계들에서 리용되고 있다. 평판현시장치에는 플라즈마현시판과 액정장치가 속한다. 비록 벡토르현시장치들은 질이 높은 직선을 현시하는데 사용할수 있지만 라스터현시기술의 개선으로 벡토르현시장치들은 라스터체계들에 의하여 대부분 교체되게 되였다.

다른 현시기술에는 3차원 및 립체보기체계가 속한다. 가상현실체계에는 립체모자나 표준영상현시장치가 들어 갈수 있다.

도형을 입력하는데 선택하여 쓸수 있는 여러가지 장치들이 있다. 건반, 누름단추통 그리고 다이얄

들은 본문, 자료값, 혹은 프로그람작성의 선택항목들을 입력하는데 사용한다. 가장 일반적인 지시장치는 마우스이지만 추적볼, 공간볼, 조종간, 유표조종건 그리고 손가락굴리개들도 화면에서 유표의 위치를 지정하는데 사용한다. 가상현실환경에서는 자료장갑이 일반적으로 사용된다. 기타 입력장치들에는 화상스캐너, 수자화기, 다침판, 빛펜 그리고 음성체계들이 속한다.

도형처리워크스테이션을 위한 경복사장치들로는 표준인쇄기와 작도기가 있으며 환등필림, 투영필림을 만들거나 영화필림으로 출력하기 위한 장치들이 보충적으로 속한다. 인쇄하는 방법에는 점행렬식, 레이자, 잉크분사식, 정전기식, 전열방법들이 있다. 작도방법에는 펜작도, 인쇄와 작도를 결합한 장치들이 속한다.

도형처리쏘프트웨어는 대체적으로 응용프로그람패키지와 프로그람작성패키지로 분류할수 있다. 응용도형처리쏘프트웨어에는 CAD프로그람, 도면 및 그림그리기프로그람, 그라프그리기프로그람, 가시화프로그람들이 속한다. 일반도형처리작성패키지들에는 PHIGS, PHIGS+, GKS, 3D GKS, 그리고 GL이 속한다. PHIGS, GKS, CGI 그리고 CGM과 같은 쏘프트웨어표준들은 계속 개선되고 있으며 여러가지 콤퓨터들에서 널리 사용할수 있게 되여 있다.

일반적으로 도형처리프로그람들은 직각자리표계로 자리표값들을 지적해 줄것을 요구한다. 장면안의 매개 물체는 개별적인 모형화직각자리표계에서 정의되며 후에 그것은 세계자리표로 넘겨 져 장면에 들어 간다. 세계자리표로부터 물체들은 정규화된 장치자리표로 변환되며 마지막에 현시장치자리표로 변환된다. 모형화자리표로부터 정규화된 장치자리표로의 변환은 응용에서 사용될수 있는 개별적인 장치들과는 독립적이다. 그후 장치구동기들이 정규화된 자리표를 웅근수장치자리표로 변환하는데 사용된다.

도형처리프로그람작성패키지에서 함수들은 다음과 같은 부류로 나눌수 있다. 즉 출력기초요소, 속성, 기하학적 및 모형화변환, 보기변환, 구조물조작, 입력함수들과 조종조작으로 나눌수 있다.

PHIGS와 GKS와 같은 일부 도형처리체계들은 개별적인 응용의 입출력에 쓰이는 장치나 쏘프트웨어를 지적하기 위하여 워크스테이션이라는 개념을 사용한다. 이런 체계들에서 워크스테이션식별자는 하나의 파일이나 라스터현시장치와 같은 하나의 장치 혹은 현시장치, 건반, 마우스와 같은 장치들의 결합을 가리킬수 있다. 하나의 도형처리응용에서는 입력과 출력을 위하여 여러개의 워크스테이션을 열어 놓을수 있다.

참고문헌

평판현시장치들을 비롯하여 전자현시장치들에 대한 일반적해설은 Sherr(1993)에서 찾아 볼수 있다. 평판현시장치들을 Depp 와 Howard(1993)에서 설명되고 있다. Tannas(1985)는 평판현시장치와 CRT에 대한 참고문헌들을 제공하고 있다. 라스터도형처리장치구성에 대한 보통정보는 Foley(1990)들에서 찾아 볼수 있다. 3차원말단장치들은 Fuchs(1982)들과 Johnson(1982) 그리고 Ikedo(1984)가 설명하고 있다. 모자에 설치된 현시장치와 가상현상환경은 Chung(1989)에서 설명되고 있다. PHIGS와 PHIGS+에 대한 정보는 Hopgood와 Duce(1991), Howard(1991)들, Gaskins(1992) 그리고 Blake(1993)를 찾아 보시오. 2차원GKS표준과 도형처리표준의 개선에 대한 정보는 Hopgood(1983)들에서 얻을수 있다. GKS에 대한 보충적인 참고문헌은 Enderle, Kensg 그리고 Pfaff(1984)에 있다.

련습문제

2-1. 다음의 현시기술들의 동작특징들을 지적하시오: 라스터재생체계, 벡토르재생체계, 플라

즈마평판, 액정현시장치.

2-2. 련습문제 1의 매 현시기술에 대하여 적합한 응용을 몇가지 지적하시오.

2-3. 현재 사용하고 있는 체계의 영상현시장치에서 x와 y방향의 분해능(센치메터당 화소)을 확인하시오.

2-4. 분해능이 각각 640 × 480, 1280 × 1024, 2560 × 2048 인 세가지 서로 다른 라스터체계가 있다. 화소당 12bit 씩 기억시키려면 매 체계에서 얼마만한 크기의 프레임완충기(바이트로)가 필요한가? 만약 화소당 24bit 씩 기억시킨다면 얼마만한 기억기가 요구되는가?

2-5. 화면의 크기가 8×10in, 매 방향에서 인치당 100화소의 분해능을 가지게 설계된 RGB라스터체계가 있다. 만약 프레임완충기에서 화소당 6bit씩 기억시키려고 하면 얼마만한 기억기(바이트로)가 프레임완충기에 필요한가?

2-6. 만약 초당 10^5bit 씩 전송할수 있다면 화소당 12bit 를 가지는 640 × 480 프레임완충기를 적재하는데 얼마나 오래 걸리겠는가? 같은 전송속도로 1280 × 1024 의 분해능을 가지는 프레임완충기를 화소당 24 bit 로 적재하는데는 얼마나 오래 걸리겠는가?

2-7. 한 단어의 길이가 32bit이고 전송속도가 1mip(초당 100만개 명령)인 콤퓨터가 있다. 페지의 크기가 8.5in, 11in 인 300dpi(인치당 점) 레이자인쇄기의 프레임완충기를 채우는데 시간이 얼마나 걸리겠는가?

2-8. 분해능이 각각 640×480, 1280×1024인 두개의 라스터체계가 있다. 초당 60프레임속도로 화면을 재생하는 현시조종기로 매 체계에서 초당 얼마만한 화소들을 호출할수 있는가?

2-9. 너비 12in, 높이 9.6in의 현시구역을 가지는 영상조종기가 있다. 만약 분해능이 1280×1024이고 종횡비가 1이라면 화면에서 매점의 직경은 얼마인가?

2-10. 분해능이 1280 × 1024, 재생속도 초당 60 프레임인 라스터체계에서 화면재생기간에 매행의 화소들을 주사하는데 소모되는 시간은 얼마인가?

2-11. 분해능이 n×m(m은 주사선, n은 주사선당 화소), 재생속도 초당 r프레임, 수평귀선시간 $t_{수평}$, 수직귀선시간 $t_{수직}$인 비간격식라스터현시장치가 있다. 프레임당 총 재생시간에서 전자속의 귀선에 소모되는 부분은 얼마인가?

2-12. 분해능이 1280 × 1024, 재생속도 60Hz, 수평귀선시간 5 ㎲, 수직귀선시간 500㎲인 비간격식라스터체계가 있다. 프레임당 총 재생시간에서 전자속의 귀선에 소모되는 부분은 얼마인가?

2-13. 어떤 완전색(화소당 24bit)RGB라스터체계가 512 × 512 프레임완충기를 가지고 있다고 하자. 얼마만한 서로 다른 색(세기준위)을 선택하여 쓸수 있는가? 한번에 얼마만한 각이한 색들을 현시할수 있는가?

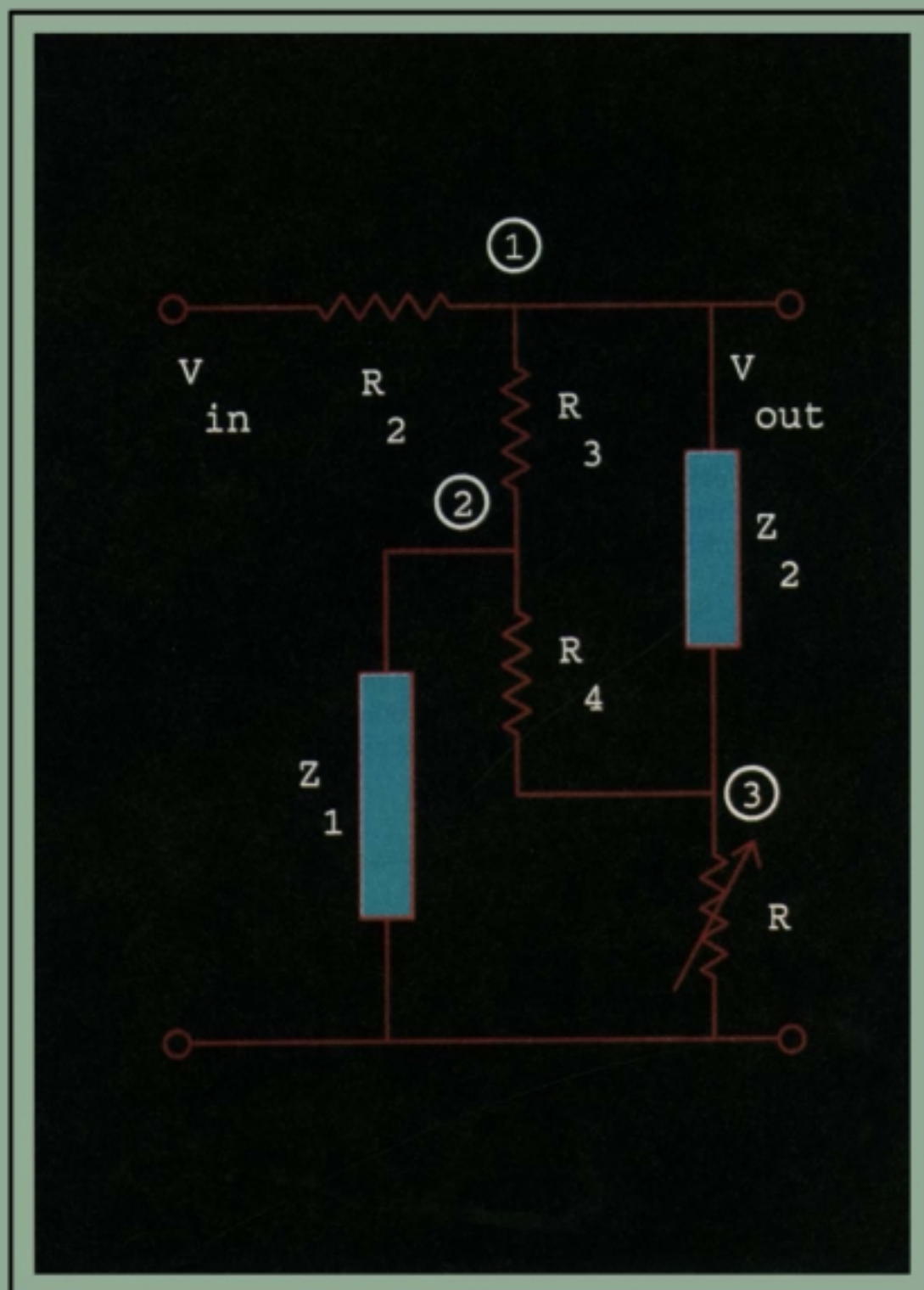
2-14. 가변초점거울을 립체현시체계와 함께 쓰는 3 차원현시장치의 우결함을 말하시오.

2-15. 가상현실체계에서 일반적으로 사용되는 여러가지 입출력구성요소들을 들어 보시오. 또한 사용자들이 2차원 및 립체현시장치와 같은 여러가지 출력장치들에 현시된 가상장면과 어떻게 대화하는가를 설명하시오.

2-16. 가상현실체계가 설계에서 어떻게 리용될수 있는가를 설명하시오. 가상현실체계의 기타 다른 응용실례를 들어 보시오.

2-17. 대형화면현시장치의 몇가지 응용을 들어 보시오.

2-18. 프로그람작성자들을 위하여 설계된 일반도형처리체계와 건축설계와 같은 전문응용을 위하여 설계된것들사이에 어떤 차이가 있는지 설명하시오.

3장. 출력기초요소

그림은 여러가지 방법으로 표현할수 있다. 라스터현시장치인 경우에는 현시장치의 매 화소위치에서의 세기의 모임으로 그림이 완전히 표현된다. 한편 그림을 장면안의 지적된 자리표위치들에 놓이는 나무와 지형, 가구와 벽과 같은 복잡한 물체들의 모임으로 서술할수도 있다. 물체의 형태 및 색은 내부적으로 화소배렬 또는 선분, 다각형색구역과 같은 기초적인 기하학적구조물들의 모임으로 표현할수 있다. 이때 장면은 화소배렬을 프레임완충기에 적재하거나 기초적인 기하학적구조물서술을 화소무늬로 주사변환함으로써 현시되게 된다. 일반적으로 도형처리프로그람들은 장면을 **출력기초요소**(output primitives)라고 하는 이런 기초적인 기하학적구조물들에 의하여 표현하며 출력기초요소들의 모임을 보다 복잡한 구조물덩어리로 만드는 함수들을 제공하여 준다. 매개 출력기초요소들은 입력자리표자료와 물체가 현시될 방식에 대한 기타 정보에 의하여 지적된다. 점과 선분은 그림의 제일 간단한 기하학적요소이다. 그림을 만드는데 리용될수 있는 추가적인 출력기초요소들에는 원과 기타 원추곡선, 2차곡면, 스플라인곡선과 곡면, 다각형색구역, 문자렬들이 속한다. 이 장에서는 그림생성절차에 대한 설명을 2차원출력기초요소들을 현시하기 위한 장치준위에서의 알고리듬들을 고찰하는것으로부터 시작한다. 그러면서 라스터도형처리체계에서의 주사변환방법들을 하나하나 해설한다. 또한 출력함수들이 도형처리프로그람들에서 어떻게 주어 질수 있는가를 고찰하고 PHIGS언어에서 사용가능한 출력함수들을 살펴 본다.

1절. 점과 직선

점찍기는 응용프로그람에서 제공되는 하나의 자리표위치를 사용하는 출력장치의 해당한 조작으로 변환함으로써 수행된다. 실례로 CRT현시장치에서는 지적된 위치에서 화면의 형광체가 발광하도록 전자속을 편향시킨다. 전자속이 어떻게 위치를 정하는가는 현시장치기술에 관계된다. 우연주사(벡토르)체계에서는 점찍기명령을 현시목록에 기억시켜 놓고 있는데 이 명령안의 자리표값들은 매개 재생주기에 현시할 화면위치에 전자속을 보내는 편향전압으로 변환된다. 한편 흑백라스터체계에서는 점을 프레임완충기안의 지정된 화면위치에 대응하는 비트값을 1로 설정하여 현시한다. 그다음 전자속이 매개 수평주사선을 따라 훑을 때 프레임완충기에서 값 1을 만들 때마다 전자폭발을 일으키게 한다(점을 현시한다). RGB체계에서는 프레임완충기에 화면의 화소위치에 현시할 세기에 대한 색코드가 적재된다.

직선은 직선경로를 따라 주어 진 두개의 끝점사이의 중간위치들을 계산하여 그린다. 출력장치에는 끝점들사이의 이 위치들을 채우도록 지시한다. 벡토르펜작도기, 우연주사현시장치와 같은 상사장치들은 직선을 한 끝점으로부터 다른 끝점까지 원활하게 그릴수 있다. 원활한 선을 만드는데 요구되는 x 및 y방향에서의 변화에 비례하여 선형적으로 변하는 수평 및 수직편향전압이 발생된다.

수자장치들에서는 두 끝점사이의 불련속점들을 현시하여 선분을 현시한다. 선경로에 따르는 불련속자리표위치들은 선의 방정식으로부터 계산한다. 이때 라스터영상현시장치에서는 선의 색(세기)이 프레임완충기의 대응하는 화소자리표에 적재된다. 그러면 영상조종기가 프레임완충기를 읽으면서 화면화소들을 《현시》한다. 화면위치는 옹근수값으로 지적되므로 현시되는 위치들은 주어 진 두 끝점사이의 실제선의 위치에 대한 근사화에 불과하다. 실례로 계산된 선위치(10.48, 20.51)는 화소위치(10, 21)로 변환되게 된다. 이 자리표값을 옹근수로 둥그리기하면 선이 그림 3-1에 현시된바와 같이 계단모형으로(터실터실하게) 현시된다. 라스터직선에서 독특한 이 계단모양은 저분해능체계에서는 하나하나 나타나지만 그것을 고분해능체계에서 현시하면 잘 나타나지 않는다. 라스터직선을 원활

하게 하는 보다 효과적인 기술들은 선경로를 따라 화소세기를 조정하는데 기초하고 있다.

이 장에서 설명되는 라스터도형처리장치준위알고리듬들에서 물체위치는 직접 옹근수장치자리표로 지적된다. 당분간 우리는 화소위치를 주사선번호와 렬번호(주사선에 따르는 화소위치)로 본다. 그림 3-2에서 이 주소화개념을 설명한다. 주사선은 화면의 밑에서 시작하여 0부터 련속적으로 번호가 매겨 지며 화소렬은 매개 주사선을 따라서 왼쪽에서 오른쪽으로 0부터 번호가 매겨 진다. 10절에서 다른 또 하나의 화소주소화개념을 고찰한다.

그림 3-1. 직선이 화소위치들의 련속으로 발생될 때 만들어 지는 계단효과(터실터실한것)

주사선
번호

5
4
3
2
1
0

0 1 2 3 4 5 화소렬번호

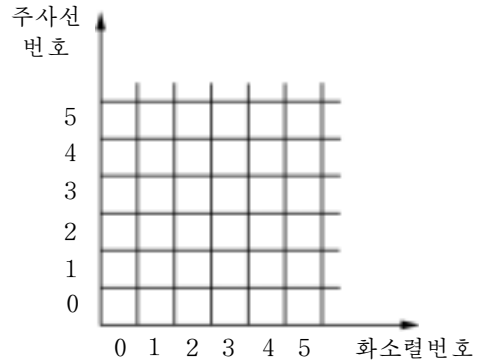



그림 3-2. 주사선번호와 렬번호에 의하여 주어 지는 화소위치

앞으로 y번째 주사선과 x번째 렬에 대응하는 프레임완충기의 위치에 지적된 색을 적재하기 위한

```
setPixel(x, y)
```

형식의 사용가능한 저수준함수를 가지고 있다고 가정한다. 또한 때때로 지적된 위치에 대한 현재 프레임완충기의 세기설정을 볼 필요도 있으므로 이것은 저수준함수

```
getPixel(x, y)
```

에 의하여 수행된다고 가정한다.

2절. 직선그리기알고리듬

방향결수에 의한 직선방정식은

$$y = m \cdot x + b \tag{3-1}$$

이다. 여기서 m은 직선의 경사도를 표현하며 b는 y축자르기값이다. 그림 3-3에 보여 준바와 같이 선분의 두 끝점이 위치 (x_1, y_1), (x_2, y_2)로 주어 지면 다음의 계산에 의하여 경사도 m과 y축자르기 b에 대한 값을 결정할수 있다.

$$m = \frac{y_2 - y_1}{x_2 - x_1} \tag{3-2}$$

$$b = y_1 - m \cdot x_1 \tag{3-3}$$

직선현시알고리듬들은 직선의 방정식 3-1과 식 3-2 및 3-3에서 주어 지는 계산에 기초한다.

선을 따라 임의의 주어 진 x의 간격 Δx에 대응하는 y 간격 Δy는 식 3-2로부터

$$\Delta y = m\,\Delta x \tag{3-4}$$

와 같이 계산할수 있다. 류사하게 주어 진 Δy에 대응하는 x간격 Δx를

$$\Delta x = \frac{\Delta y}{m} \tag{3-5}$$

와 같이 얻을수 있다. 상사장치들에서는 이 식들이 편향전압을 결정하기 위한 기초를 준다. 경사도의 절대값이 $|m|<1$인 직선에서는 Δx 를 미소수평편향전압에 비례되게 설정할수 있다. 그러면 대응하는 수직편향전압은 식 3-4로부터 계산되는 Δy 에 비례되게 설정하게 된다. 경사도절대값이 $|m|>1$인 직선에서는 Δy 를 미소수직편향전압에 비례되게 설정한다. 그러면 대응하는 수평편향전압은 식 3-5로부터 계산되는 Δx에 비례되게 설정한다. $m=1$인 직선에서는 $\Delta x=\Delta y$이며 수평 및 수직편향전압들이 같다. 매개 경우에 경사도가 m인 원활한 직선이 주어 진 끝점들사이에서 발생된다.

라스터체계에서는 직선이 화소들로 현시되며 수평 및 수직방향의 걸음크기는 화소간격에 의하여 제한된다. 즉 직선을 불련속위치들에서 표본화하고 매개 표본위치들에서 직선에 제일 가까운 화소들을 결정하게 된다. 직선에 대한 이 주사변환처리를 그림 3-4에서는 x축을 따라 불련속표본위치를 가지는 거의 수평인 선에 대하여 설명하였다.

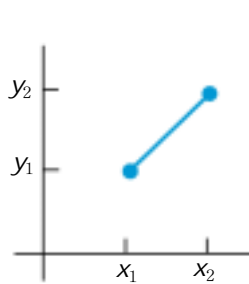



그림 3-3. 끝점위치 (x_1, y_1)와 (x_2, y_2)사이의 선경로

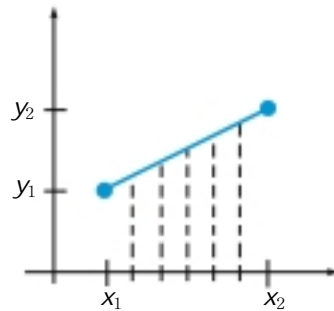



그림 3-4. x축을 따라 x_1과 x_2사이에 5개의 표본위치를 가지는 선분

DDA 알고리듬

수자미분분석기(digital differential analyzer, DDA)는 식 3-4, 3-5를 리용하여 Δy 또는 Δx를 계산하는데 기초한 주사변환직선알고리듬이다.

어느 한 자리표에서는 단위간격으로 직선을 표본화하고 다른 자리표에 대하여서는 선경로에 제일 가까운 대응하는 옹근수값을 결정한다.

그림 3-3에 보여 준바와 같이 먼저 정의경사도를 가지는 직선을 고찰하자. 경사도의 절대값이 1보다 작거나 같으면 단위 x간격($\Delta x=1$)으로 표본화하고 매개 린접된 y값은

$$y_{k+1} = y_k + m \tag{3-6}$$

와 같이 계산한다. 첨수 k는 첫 점에 대하여 1부터 시작하는 옹근수값을 가지며 마지막끝점에 도달할 때까지 1씩 증가한다. m은 0과 1사이의 임의의 실수일수 있기때문에 계산된 y값은 제일 가까운 옹근수로 둥그려야 한다.

1보다 큰 정의경사도를 가지는 직선에 대하여서는 x와 y의 역할을 반대로 한다. 즉 단위 y간격($\Delta y=1$)으로 표본화하고 린접되는 x값은

$$x_{k+1} = x_k + \frac{1}{m} \tag{3-7}$$

와 같이 계산한다.

식 3-6과 3-7에서는 직선이 왼쪽 끝점으로부터 오른쪽 끝점에로 가면서 처리된다는 가정에 기초하고 있다(그림 3-3). 시작끝점이 오른쪽에 있도록 이 처리를 반대로 한다면 Δx=-1과

$$y_{k+1} = y_k - m \tag{3-8}$$

또는 Δy=-1(경사도가 1보다 클 때)과

$$x_{k+1} = x_k - \frac{1}{m} \tag{3-9}$$

를 얻게 된다.

식 3-6 ~ 3-9는 또한 부의 경사도를 가지는 직선을 따라 화소위치를 계산하는데도 리용할수 있다. 경사도의 절대값이 1보다 작고 시작끝점이 왼쪽에 있으면 Δx =1로 설정하고 y값은 식 3-6에 의하여 계산한다. 시작끝점이 오른쪽에 있을 때는(동일한 경사도에 대하여) Δx = -1로 설정하고 y위치는 식 3-8로부터 얻는다. 류사하게 부의 경사도의 절대값이 1보다 클 때에는 Δy = -1과 식 3-9 또는 Δy=1과 식 3-7을 리용한다.

이 알고리듬은 다음의 프로그람에 개괄되여 있는데 입력으로서 두개 끝점의 화소위치를 받아 들인다. 끝점위치들사이의 수평 및 수직편차는 파라메터 dx와 dy에 할당된다. 그중 보다 큰 절대값을 가지는 편차가 파라메터 steps의 값을 결정한다. 화소위치 (x_a, y_a)로부터 시작하여 직선경로를 따라 다음화소위치를 발생시키는데 필요한 매 걸음에서의 편위를 결정한다. 이 처리를 steps값만큼 반복한다. 만일 dx의 크기가 dy의 크기보다 크고 xa가 xb보다 작으면 x와 y방향에서 증분값은 각각 1과 m이다. 만일 x방향에서의 변화가 더 크지만 xa가 xb보다 크면 감소량 -1과 $-m$이 직선우의 매개 새로운 점을 발생시키는데 리용된다. 그렇지 않은 경우는 y방향에서 단위증가(감소), x방향에서 $1/m$만한 증가(감소)를 리용한다.

```
#include "device. h"

#define ROUND(a) ((int)(a+0.5))

void lineDDA (int xa, int ya, int xb, int yb)
{
   int dx = xb - xa, dy = yb - ya, steps, k;
   float xIncrement, yIncrement, x = xa, y = ya;

   if (abs (dx) > abs (dy)) steps = abs (dx) ;
   else steps = abs (dy) ;
   xIncrement = dx / (float) steps;
   yIncrement = dy / (float) steps;

   setPixel (ROUND(x), ROUND(y));
   for (k=0; k<steps; k++) {
    x += xIncrement;
    y += yIncrement;
    setPixel (ROUND(x), ROUND(y));
  }
}
```

DDA알고리듬은 식 3-1을 직접 리용하는것보다 화소위치를 계산하는데서 더 빠른 방법이다. 그것은 라스터특성을 리용하여 식 3-1에서의 곱하기를 없애고 적당한 증분량을 x 또는 y방향에서의 선경로에 따르는 화소위치의 걸음에 적용하고 있다. 그러나 류점수증분의 련속적인 더하기에서 둥그리기오차의 축적은 긴 선분에서 계산된 화소의 위치가 실제직선경로로부터 많이 벗어 날수 있다. 게다가 프로그람 lineDDA에서의 둥그리기조작과 류점수계산에는 여전히 시간이 소비된다. 모든 계산이 옹근수조작으로 간단해 지도록 증분 m과 $1/m$을 옹근수부와 소수부로 가름으로써 DDA알고리듬의 실행을 개선할수 있다. 옹근수걸음에서의 $1/m$증분의 계산방법은 3장 11절에서 설명된다. 다음절에서 직선과 곡선에 다같이 적용될수 있는 있는 보다 일반적인 주사선절차를 고찰한다.

브리센함의 직선알고리듬

브리센함(Bresenham)에 의하여 개발된 정확하고도 효률적인 라스터직선발생알고리듬은 증분의 옹근수계산만을 리용하여 직선을 주사변환한다. 이 방법은 원과 기타 곡선들을 현시하는데도 적응시킬수 있다. 그림 3-5와 3-6은 선분이 그려 질 현시장치화면의 부분을 표현하고 있다. 수직축은 주사선위치를 보여 주며 수평축은 화소렬을 표현한다. 이 실례들에서 단위 x간격으로 표본화하면 매개 표본걸음에서 두개의 가능한 화소위치중 어느것이 직선경로에 더 가까운가를 결정하야 한다. 그림 3-5에 보여 준 왼쪽 끝점으로부터 시작하면 다음의 표본위치에서 위치 (11, 11)이나 혹은 (11, 12)중 어느 화소를 현시하겠는가를 결정하여야 한다. 류사하게 그림 3-6 은 화소위치 (50, 50)의 왼쪽 끝점으로부터 시작하는 부의경사도직선의 경로를 보여 준다. 여기서는 다음화소위치를 (51, 50)과 (51, 49)중 어느것을 선택하겠는가가 제기된다. 이 질문은 브리센함의 직선알고리듬에서 어떤 옹근수파라메터의 부호를 검사하면 대답이 얻어 지게 되는데 그 값은 실지 직선경로로부터의 두 화소위치의 편차들사이의 차에 비례한다.

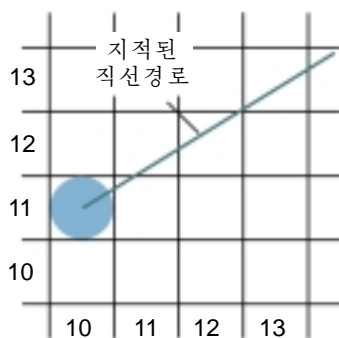



그림 3-5. 선분이 주사선 11의 렬 10에 있는 화소로부터 시작하여 그려 지는 현시장치화면의 부분

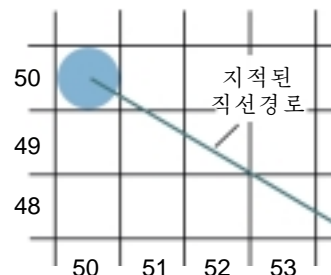



그림 3-6. 부의 경사도선분이 주사선 50의 렬 50에 있는 화소로부터 시작하여 그려 지는 현시장치화면의 부분

브리센함의 방법을 설명하기 위하여 먼저 1보다 작은 정의경사도를 가지는 직선에 대한 주사변환처리를 고찰하자. 직선경로에 따르는 화소위치는 단위 x간격으로 표본화하여 결정한다. 주어 진 직선의 왼쪽 끝점 (x_0, y_0)으로부터 시작하여 매개 린접된 렬(x위치)로 가면서 주사선의 y값이 직선경로에 제일 가까운 화소를 현시한다. 그림 3-7은 이 처리에서의 k번째 걸음을 보여 준다. (x_k, y_k)의 화소가 현시된다고 하면 그다음에는 렬 x_k+1에서 어느 화소를 현시하겠는가를 결심하여야 한다. 선택할수 있는것은 위치 (x_k+1, y_k)와 (x_k+1, y_k+1)에 있는 화소이다.

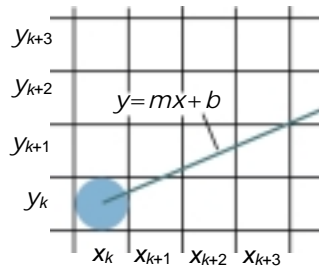



그림 3-7. 경사도가 $0<m<1$ 인 선분의 경로를 따라 현시될 주사선 y_k렬 x_k에 있는 화소를 보여 주는 화면살창의 부분

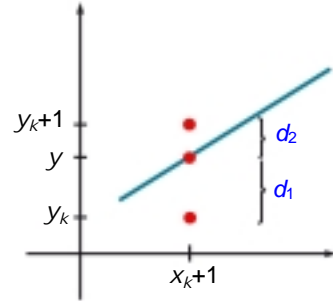



그림 3-8. 표본위치 x_k+1 에서 화소들의 위치와 선의 y자리표 사이의 거리

표본위치 x_k +1에서 수학적인 직선경로로부터의 수직화소편차를 d_1과 d_2로 표시하자(그림 3-8).. 화소렬위치 x_k+1에서 수학적직선의 y자리표는

$$y = m(x_k + 1) + b \tag{3-10}$$

와 같이 계산된다. 그러면

$$\begin{aligned} d_1 &= y - y_k \\ &= m(x_k + 1) + b - y_k \end{aligned}$$

이고

$$\begin{aligned} d_2 &= (y_k + 1) - y \\ &= y_k + 1 - m(x_k + 1) - b \end{aligned}$$

이 두 편차사이의 차는

$$d_1 - d_2 = 2m(x_k + 1) - 2y_k + 2b - 1 \tag{3-11}$$

이다.

직선알고리듬에서 k번째 걸음에 대한 결심파라메터 p_k는 식 3-11을 옹근수계산만을 포함하도록 재정돈하면 얻을수 있다. 이것은 $m=\Delta y/\Delta x$(여기서 Δy 및 Δx는 끝점위치들의 수직 및 수평편차이다.)로 바꾸고

$$\begin{aligned} p_k &= \Delta x(d_1 - d_2) \\ &= 2\Delta y \cdot x_k - 2\Delta x \cdot y_k + c \end{aligned} \tag{3-12}$$

로 정의하는것에 의하여 실현한다.

p_k의 부호는 이 실례에서 $\Delta x>0$ 이므로 $d_1 - d_2$의 부호와 같다. 파라메터 c는 상수로서 값 $2\Delta y+\Delta x(2b-1)$을 가진다. 이것은 화소위치에 무관계하며 p_k재귀계산에서는 제외된다. 만약 y_k에서의 화소가 y_k+1에서의 화소보다 직선경로에 더 가깝다면(즉 $d_1 < d_2$) 결심파라메터 p_k 는 부이다. 이 경우 아래화소를 현시하고 그렇지 않으면 웃화소를 현시한다.

직선우에서 자리표변화는 x 및 y방향에서 다같이 단위걸음으로 일어 난다. 따라서 린접한 결심파라메터의 값을 증분의 옹근수계산을 리용하여 얻을수 있다. k +1번째 걸음에서 결심파라메터는 식 3-12로부터

$$p_{k+1} = 2\Delta y \cdot x_{k+1} - 2\Delta x \cdot y_{k+1} + c$$

와 같이 평가된다. 식 3-12 를 웃식에서 덜면

$$p_{k+1} - p_k = 2\Delta y(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$$

을 얻는다. 그런데 $x_{k+1} = x_k+1$ 이므로

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k) \qquad (3-13)$$

여기서 $y_{k+1}-y_k$항은 파라메터 p_k의 부호에 따라 0 또는 1이다.

결심파라메터의 이 재귀적인 계산은 직선의 왼쪽 끝점자리표에서 시작하여 매개 옹근수 x위치에서 진행된다. 첫번째 파라메터 p_0은 시작화소의 위치 (x_0, y_0)에서 식 3-12와 $\Delta y/\Delta x$로 계산되는 m에 의하여 평가된다.

$$p_0 = 2\Delta y - \Delta x \qquad (3-14)$$

1보다 작은 정의경사도를 가지는 직선에 대한 브리센함의 직선그리기를 아래 표의 걸음들에서 개괄할수 있다. 상수 $2\Delta y$와 $2\Delta y - 2\Delta x$는 주사변환될 직선에서 한번만 계산된다. 그러므로 산수계산은 이 두 상수의 옹근수더하기 및 덜기만을 포함한다.

$|m| < 1$일 때의 브리센함의 직선그리기알고리듬

1. 직선의 두 끝점을 입력하고 왼쪽 끝점을 (x_0, y_0)에 기억시킨다.
2. (x_0, y_0)을 프레임완충기에 적재한다. 즉 첫 점을 현시한다.
3. 상수 Δx, Δy, $2\Delta y$와 $2\Delta y - 2\Delta x$를 계산하고 결심파라메터의 시작값을 다음과 같이 얻는다.

$$p_0 = 2\Delta y - \Delta x$$

4. $k = 0$에서 시작하여 직선을 따라 매 x_k에서 다음의 검사를 진행한다.
   만일 $p_k < 0$이면 현시할 다음점은 (x_k+1, y_k)이며

$$p_{k+1} = p_k + 2\Delta y$$

   그렇지 않으면 현시할 다음점은 (x_k+1, y_k+1)이며

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$

5. 걸음 4를 Δx번 반복한다.

실례 3-1. 브리센함의 직선그리기

알고리듬을 설명하기 위하여 끝점 (20, 10)과 (30, 18)을 가지는 직선을 수자화하자. 이 직선은 경사도가 0.8이며

$$\Delta x = 10, \qquad \Delta y = 8$$

초기결심파라메터는 값

$$p_0 = 2\Delta y - \Delta x$$
$$= 6$$

을 가진다. 린접한 결심파라메터들의 계산을 위한 증분량은

$$2\Delta y = 16, \qquad 2\Delta y - 2\Delta x = -4$$

이다. 초기점 $(x_0, y_0) = (20, 10)$을 현시하고 직선경로를 따라 린접한 화소위치들을 결심파라메터로부터 다음과 같이 결정한다.

k	p_k	(x_{k+1}, y_{k+1})	k	p_k	(x_{k+1}, y_{k+1})
0	6	(21, 11)	5	6	(26, 15)
1	2	(22, 12)	6	2	(27, 16)
2	-2	(23, 12)	7	-2	(28, 16)
3	14	(24, 13)	8	14	(29, 17)
4	10	(25, 14)	9	10	(30, 18)

선경로를 따라 발생되는 화소현시를 그림 3-9 에 보여 주었다.

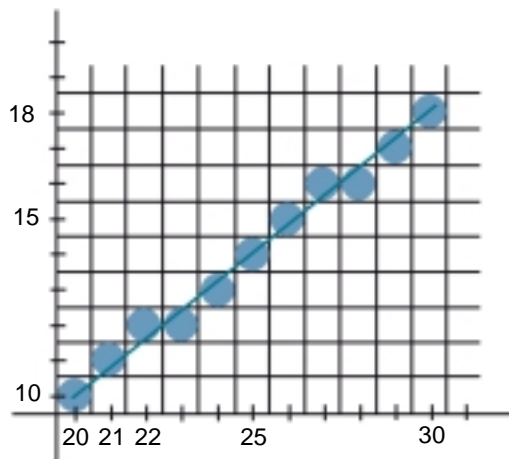



그림 3-9. 끝점 (20, 10)과 (30, 18)사이의 직선경로를 따르는 화소위치, 브리센함의 직선알고리듬으로 현시되였다.

0<m<1 범위의 경사도에 대한 브리센함의 직선그리기의 실현을 다음프로그람에 주었다. 직선의 끝점화소위치들이 이 프로그람에 넘겨 지며 화소들은 왼쪽 끝점으로부터 시작하여 오른쪽 끝점으로 현시된다. setPixel 을 호출하면 기정색값을 프레임완충기의 주어 진 (x, y)화소위치에 적재한다.

```
#include "device. h"

void lineBres (int xa, int ya, int xb, int yb)
{
  int dx = abs (xa - xb), dy = abs (ya - yb) ;
  int p = 2 * dy - dx;
  int twoDy = 2 * dy, twoDyDx = 2 * (dy - dx) ;
  int x, y, xEnd;

  /* Determine which point to use as start, which as end */
  if (xa > xb) {
   x = xb ;
   y = yb;
   xEnd = xa ;
  }
```

```
    else {
      x = xa ;
      y = ya;
      xEnd = xb ;
    }
    setPixel (x, y) ;

    while (x < xEnd) {
      x++;
      if (P < 0)
         p += twoDy;
      else {
         y++;
         p += twoDyDx;
      }
      setPixel (x, y) ;
    }
  }
```

브리센함의 알고리듬은 xy평면의 서로 다른 8분구 및 4분구들사이의 대칭성을 고려하여 임의의 경사도를 가지는 직선으로 일반화한다. 1보다 큰 정의경사도를 가지는 직선에 대하여서는 x 및 y방향의 역할을 바꾼다. 즉 y방향을 따라 단위걸음으로 나가면서 직선경로에 제일 가까운 린접한 x값들을 계산한다. 또한 어느 끝점으로부터 시작하여도 화소를 현시할수 있도록 프로그람을 교정할수도 있다. 정의경사도를 가지는 직선에서 초기위치가 오른쪽 끝점이라면 오른쪽에서 왼쪽으로 갈 때 x와 y는 다 같이 감소한다. 시작끝점에 관계없이 같은 화소가 현시된다는것을 보증하기 위하여 직선경로에 대한 두개의 수직편차가 같을 때마다($d_1 = d_2$) 두개의 후보화소중 항상 우(또는 아래)를 선택한다. 부의경사도에서는 한 자리표가 증가할 때 다른것은 감소한다는것을 제외하면 절차는 류사하다. 마지막으로 특수한 경우들은 따로따로 처리할수 있다. 수평선($\Delta y = 0$), 수직선($\Delta x = 0$), $|\Delta x| = |\Delta y|$ 인 대각선은 직선그리기알고리듬을 통하여 처리하지 않고 프레임완충기에 직접 적재할수 있다.

직선의 병렬알고리듬

이때까지 설명한 직선발생알고리듬에서는 화소위치를 순차적으로 결정한다. 병렬콤퓨터에서는 직선경로에 따르는 화소위치들을 사용가능한 여러 처리기들에 계산을 분담시켜 놓는 방법으로 계산을 동시에 진행할수 있다. 문제를 분할하는 한가지 방법은 이미 있는 순차알고리듬을 다중처리기의 우점을 살리도록 적응시키는것이다. 다른 하나는 화소위치들이 병렬적으로 효률적으로 계산될수 있도록 처리를 설정하는것이다. 병렬알고리듬창안에서 중요한것은 사용가능한 처리소자들사이에서 처리부담의 균형을 맞추는것이다.

n_p개의 처리기가 주어 지면 직선경로를 n_p개의 구획들로 부분분할하고 매 부분구획에서 선분들을 동시에 발생시키는 방법으로 병렬브리센함직선알고리듬을 만들수 있다. 경사도가 $0<m<1$ 이고 왼쪽끝점자리표위치가 (x_0, y_0)인 직선에 대하여서는 직선을 정의 x방향을 따라 분할해 놓는다. 린접한 구획의 시작 x위치들사이의 거리는

$$\Delta x_p = \frac{\Delta x + n_p - 1}{n_p} \qquad (3-15)$$

와 같이 계산할수 있다. 여기서 Δx 는 직선의 너비이며 Δx_p 는 구획들의 너비값으로서 옹근수나누기를 써서 계산한다. 구획과 처리기에 0, 1, 2, …, n_p-1 까지 번호를 매기면 k 번째 구획의 시작 x 자리표는

$$x_k = x_0 + k\Delta x_p \qquad (3-16)$$

와 같이 계산된다. 실례로 Δx =15이고 n_p = 4개의 처리기를 가지고 있다고 하면 구획들의 너비는 4이고 구획들의 시작 x값들은 x_0, x_0+4, x_0+8, x_0+12이다. 이 분할에서 마지막(제일 오른쪽)부분구간의 너비가 일부 경우 다른것들보다 더 작아 질수 있다. 게다가 선의 끝점이 옹근수가 아니면 절단오차로 선의 길이에 따라 구획의 너비가 변할수 있다.

구획들에 브리센함의 알고리듬을 적용하자면 매개 구획에 대한 y자리표의 초기값과 결심파라메터의 초기값이 필요하다. 매개 구획에서 y방향에서의 변화 Δy_p는 직선의 경사도 m과 구획의 너비 Δx_p 로부터 계산된다.

$$\Delta y_p = m\Delta x_p \qquad (3-17)$$

그러면 k번째 구획에서 시작 y자리표는

$$y_k = y_0 + \mathrm{round}(k\Delta y_p) \qquad (3-18)$$

k번째 부분구간의 시작점에서 브리센함알고리듬의 초기결심파라메터는 식 3-12 로부터 얻어 진다.

$$p_k = (k\Delta x_p)(2\Delta y) - \mathrm{round}(k\Delta y_p)(2\Delta x) + 2\Delta y - \Delta x \qquad (3-19)$$

이렇게 되면 매개 처리기는 부분구간에 대한 시작결심파라메터와 시작자리표 (x_k, y_k)를 리용하여 할당된 부분구간에서 화소위치들을 계산한다. 또한 시작값 y_k와 p_k에 대한 계산에서 $m = \Delta y / \Delta x$로 바꾸고 항들을 재정돈하면 류점수계산을 옹근수계산으로 간단히 할수 있다. 경사도가 1보다 큰 직선에 대한 병렬브리센함알고리듬의 확장은 직선을 y 방향에서 분할하고 부분구간의 시작 x 값을 계산하여 얻는다. 부의 경사도인 경우에는 한 방향에서는 자리표값을 증가시키고 다른 방향에서는 감소시킨다.

라스터체계에서 병렬알고리듬을 얻는 다른 방법은 매 처리기를 화면화소들의 개별적인 부분들에 할당하는것이다. 충분한 개수의 처리기(65000개이상의 처리기를 가지는 Connection Machine CM-2와 같은)를 가지고 있으면 매개 처리기를 어떤 화면구역안의 개별적인 화소들에 할당할수 있다. 이 방법에서는 개개의 처리기를 직선령역(경계직4각형)에 들어 가는 매 화소들에 할당하고 직선경로로부터 화소까지의 거리를 계산하여 직선현시에 적응시킨다. 직선의 경계직4각형안의 화소수는 $\Delta x \cdot \Delta y$이다(그림 3-10). 그림 3-10에서 직선으로부터 자리표가 (x, y)인 화소까지의 수직거리 d는 다음계산에 의하여 얻어 진다.

$$d = Ax + by + C \qquad (3-20)$$

여기서

$$A = \frac{-\Delta y}{\text{선의 길이}}$$

$$B = \frac{\Delta x}{\text{선의 길이}}$$

$$C = \frac{x_0\Delta y - y_0\Delta x}{\text{선의 길이}}$$

$$\text{선의 길이} = \sqrt{\Delta x^2 + \Delta y^2}$$

상수 A, B, C는 매 직선에 대하여 한번만 평가되며 매개 처리기는 화소거리 d를 계산하기 위하여 두번의 곱하기와 두번의 더하기를 하여야 한다. 화소는 d가 지정된 선두께파라메터보다 작으면 현시된다.

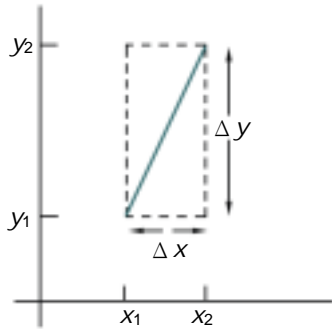



그림 3-10. 자리표너비들이 Δx와 Δy인 직선에 대한 경계직4각형

화면을 하나하나의 화소로 갈라 놓는 대신에 매개 처리기에 직선의 경사도에 따라서 주사선이나 혹은 화소들의 렬을 할당할수도 있다. 이때에 처리기는 직선과 처리기에 할당된 화소들의 수평행 또는 수직렬과의 사귐점을 계산한다. 경사도가 $|m|<1$인 직선의 경우는 직선의 방정식을 매개 처리기가 x렬값을 주고 y에 대하여 간단히 푼다. 경사도절대값이 1보다 큰 직선인 경우는 직선의 방정식을 매개 처리기가 주사선 y값을 주고 x에 대하여 푼다. 이러한 직접방법들은 비록 순차기계에서는 느리지만 다중처리기를 리용하면 대단히 능률적으로 수행될수 있다.

3절. 프레임완충기의 적재

선분 또는 다른 물체들이 라스터체계의 현시장치에서 주사변환될 때에는 프레임완충기위치들이 계산되여야 한다. 우리는 이것이 setPixel 함수에 의하여 수행된다고 가정하였는데 이 함수는 화소의 세기값을 프레임완충기 배렬안의 대응하는 주소에 기억시킨다. 주사변환알고리듬은 련속된 단위간격에서의 화소위치들을 발생시킨다. 이것은 프레임완충기주소를 계산하는데 증분방법을 리용할수 있게 한다.

구체적인 실례로 프레임완충기배렬은 행기본순서로 주소화되며 화소위치는 화면의 왼쪽 아래구석의 (0, 0)부터 오른쪽 웃구석 (x_{max}, y_{max})까지 변한다고 하자(그림 3-11). 2값체계(화소당 1bit)에서 화소위치 (x, y)에 대한 프레임완충기비트주소는

$$주소(x, y) = 주소(0, 0) + y(x_{max} + 1) + x \qquad (3-21)$$

와 같이 계산된다. 주사선을 따라 움직일 때 $(x+1,\ y)$의 화소에 대한 프레임완충기주소는 (x, y)위치에 대한 주소로부터 편위를 다음과 같이 계산할수 있다.

$$주소(x+1,\ y) = 주소(x, y) + 1 \qquad (3-22)$$

(x, y)로부터 다음주사선에로 대각선걸음할 때 $(x+1,\ y+1)$ 의 화소에 대한 프레임완충기주소는

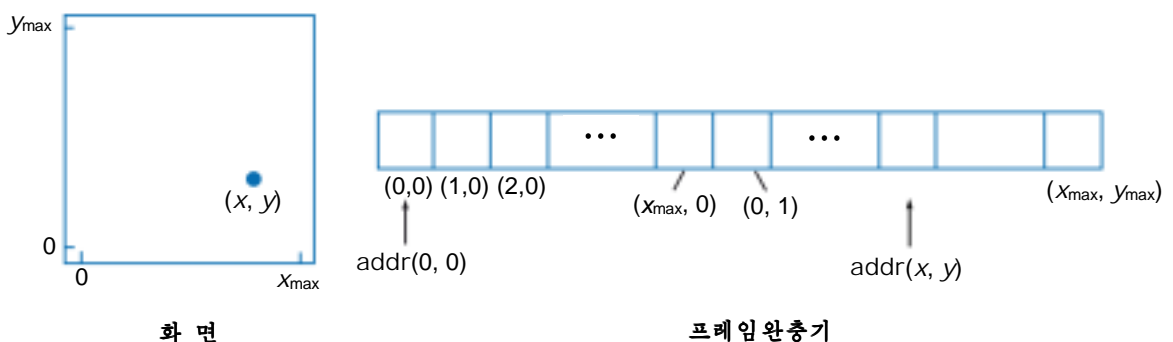
$$주소(x+1,\ y+1) = 주소(x,\ y) + x_{max} + 2 \qquad (3-23)$$

화 면 프레임완충기

그림 3-11. 프레임완충기안에서 행기본순서로 선형적으로 기억되는 화소의 화면위치

의 계산에 의하여 얻는다. 여기서 상수 $x_{max}+2$ 는 모든 선분들에 대하여 한번만 미리 계산된다. 부의 x 및 y화면방향에서의 단위걸음에 대한 류사한 증가계산은 식 3-21로부터 얻을수 있다. 이 매개 주소계산들은 단 하나의 옹근수더하기만을 포함한다.

화소의 세기값을 기억시키는 setPixel함수의 실현방법은 매개 체계의 능력과 쏘프트웨어패키지의 설계요구에 관계된다. 매개 화소에 대하여 일정한 범위의 세기값을 현시할수 있는 체계에서는 프레임완충기주소계산에 화소의 화면위치뿐아니라 화소너비(비트수)도 들어 가게 된다.

4절. 직선함수

선분들을 지적하는 함수는 서로 다른 여러가지 형식으로 설정될수 있다. PHIGS, GKS 기타 일부 프로그람들에서의 2 차원직선함수는

```
polyline(n, wcPoints)
```

이다. 여기서 파라메터 n에는 입력될 자리표위치들의 개수와 같은 옹근수값이 할당되며 wcPoints는 입력되는 선분끝점들의 세계자리표값배렬이다. 이 함수는 련결된 n-1개 선분들의 모임을 정의하는데 리용된다. 도형처리응용들에서는 련결된 선분들의 모임이 고립된 선분보다 더 자주 쓰이므로 polyline은 보다 일반적인 직선함수를 제공해 주고 있다. 하나의 선분을 현시하자면 n=2로 설정하고 두개 끌점자리표들의 x 및 y값을 wcPoints에 넣어 준다.

polyline 리용의 실례로서 다음의 명령문들은 끌점 (50, 100), (150, 250), (250, 100)을 가지는 련결된 2개 선분을 발생시킨다.

```
wcpoints[1].x=50;
wcpoints[1].y=100;
wcpoints[2].x=150;
wcpoints[2].y=250;
wcpoints[3].x=250;
wcpoints[3].y=100;
polyline (3, wcpoints)
```

polyline함수에서의 자리표는 **절대자리표값**으로 본다. 이것은 지적된 값들이 사용하는 자리표계에서의 실지점위치라는것을 의미한다.

일부 도형처리체계들은 **상대자리표**지적을 가지는 직선(또는 점)함수를 쓴다. 이 경우에 자리표값들은 마지막에 참조된 점(현재위치라고 한다.)으로부터의 편위로 본다. 실례로 위치 (3,2)가 응용프로그람에서 참조된 마지막위치라면 (2,-1)의 상대자리표지적은 절대위치 (5,1)에 대응한다. 또한 보충적인 함수가 직선루틴이 호출되기전에 현재 위치를 설정하는데 사용될수도 있다. 이런 프로그람들이 있으면 사용자는 직선명령에 한쌍의 편위값만을 지적하면 된다. 이것은 체계에 현재위치로부터 시작하여 편위에 의하여 결정되는 마지막위치까지 직선을 현시하도록 신호한다. 이때 현재위치는 이 마지막직선위치로 갱신된다. 이와 같은 프로그람들에서는 련결되는 직선들의 한개 부분을 직선명령에 의하여 하나하나씩 그린다. 일부 도형처리프로그람들은 사용자가 상대자리표나 절대자리표를 리용하여 선끌점들을 지적할수 있게 하는 선택항목들을 제공하고 있다.

Polyline함수는 먼저 일련의 자리표변환을 진행한 다음 장치준위직선그리기루틴을 차례로 호출하

여 실현한다. PHIGS에서는 입력되는 선끌점들이 실제에 있어서 모형화자리표로 지적되며 그것은 다시 세계자리표로 변환된다. 다음세계자리표는 정규자리표로 변환되고 다음에 장치자리표로 변환된다. 이 2차원자리표변환수행에 대한 세부는 6장에서 설명한다. 장치자리표에서 절선은 브리센함의 알고리듬과 같은 직선루틴을 n-1번 불러 내여 n개의 자리표점을 련결시킨다. 매개 린접호출에서는 다음 직선부분을 그리는데 필요한 두 자리표쌍을 넘겨 주는데 이때 첫번째 끌점은 앞부분의 마지막끌점이다. 끌점들에서 세기가 두번 설정되는것을 피하기 위하여 매개 토막의 마지막끌점은 현시되지 않도록 직선알고리듬을 수정할수 있다. 현시되는 물체들의 겹침을 피하는 방법은 3장 10절에서 보다 구체적으로 설명한다.

5절. 원발생알고리듬

원은 그림과 그라프에서 자주 리용되는 요소이다. 때문에 옹근원 또는 원호발생함수는 대부분의 도형처리프로그람들에 포함되여 있다. 일반적으로 원 또는 타원을 현시하는데 하나의 절차가 제공된다.

원의 성질

원은 중심위치 (x_c, y_c)로부터 주어 진 거리 r에 있는 모든 점들의 모임으로 정의된다(그림 3-12). 이 거리는 직각자리표계에서의 피다고라스정리에 의하여

$$(x - x_c)^2 + (y - y_c)^2 = r^2 \qquad (3-24)$$

와 같이 표현된다. 이 방정식을 x축을 따라 단위걸음으로 $x_c - r$부터 $x_c + r$까지 가면서 매 위치에서의 대응하는 y값을

$$y = y_c \pm \sqrt{r^2 - (x_c - x)^2} \qquad (3-25)$$

와 같이 계산하여 원주우의 점위치를 구하는데 리용할수 있다. 그러나 이것은 원발생에서 좋은 방법이 못된다. 이 방법의 한가지 문제점은 그것이 매 단계에서 계산량이 많다는것이다. 더우기 그림 3-13 에서 보여 준바와 같이 현시되는 화소위치들사이의 간격이 일정하지 않다. 원의 경사도의 절대값이 1보다 클 때마다 x와 y를 바꾸는것(y값에 따라 x값을 계산하는것)에 의하여 간격을 조정할수도 있다. 그러나 이것은 단지 알고리듬에 요구되는 계산과 처리를 증가시킬뿐이다.

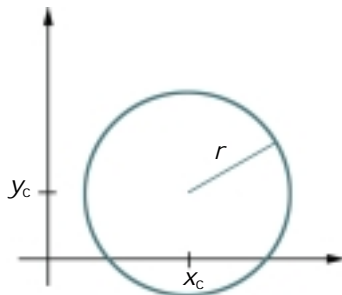



그림 3-12. 중심자리표가 (x_c, y_c), 반경 r인 원

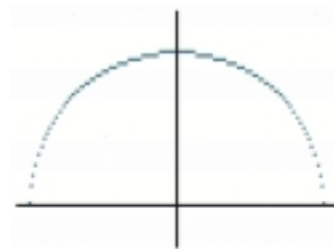



그림 3-13. 식 3-25 와 $(x_c, y_c)=(0, 0)$으로 그려 지는 원의 웃절반

그림 3-13 에 보여 준 간격이 일정하지 못한 결함을 없애기 위한 다른 방법은 원의 경계를 따라 점들을 극자리표 r 와 θ를 리용하여 계산하는것이다(그림 3-12). 원의 방정식을 보조변수극자리표형식으로 표현하면 련립방정식

$$x = x_c + r\cos\theta$$
$$y = y_c + r\sin\theta \qquad (3-26)$$

을 얻는다. 고정된 크기의 걸음각을 리용하여 이 방정식을 계산하면 화면에서 원은 원주위를 따르는 등간격점들로 현시된다. θ에 대한 걸음의 크기는 응용목적과 현시장치에 맞게 선택한다. 원주위를 따라 보다 큰 각도로 분할하는 경우에는 원경로를 근사선분들로 련결시킬수 있다. 라스터현시장치에서 보다 련속적인 경계를 만들기 위하여 걸음크기를 $1/r$로 설정할수 있다. 이것은 대략 한단위 떨어 진 화소위치들을 현시한다.

원의 대칭성을 고려하면 계산량을 줄일수 있다. 원의 형태는 매 4분구에서 같다. xy평면의 두번째 4분구에서의 부분원은 두개의 부분원들이 y축에 대하여 대칭이라는것을 고려하면 얻을수 있다. 그리고 세번째와 네번째 4분구에서의 부분원들은 첫번째 및 두번째 4분구에서의 부분원들로부터 x축에 대한 대칭성을 고려하여 얻을수 있다. 한걸음 더 나가 8분구들사이에서도 대칭이라는데 주목하자. 하나의 4분구안에 있는 린접한 8분구들에서의 부분원들은 두개의 8분구으로 나누는 45°선에 대하여 대칭이다. 이 대칭조건이 그림 3-14에서 설명된다. 여기서 1/8부채형우의 위치 $(x,\ y)$에서의 점은 xy평면의 다른 8분구들에서의 원의 7개의 점에 사상된다. 이렇게 원대칭의 우점을 살려 $x=0$부터 $x=y$까지의 부채형안의 점들만을 계산함으로써 원주우의 모든 화소위치들을 발생시킬수 있다.

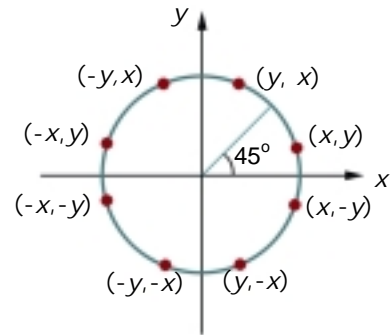



그림 3-14. 원의 대칭(한 8분구에서의 원우의 점 $(x,\ y)$의 계산은 다른 7개의 8분구들에서 보여 주는 원우의 점들을 낳는다.)

식 3-24 또는 3-26을 리용하여 원둘레를 따르는 화소위치들을 결정하는것은 많은 계산시간이 요구된다. 직각자리표계에서 방정식 3-24에는 곱하기 및 2차뿌리계산이 들어 있으며 보조변수방정식에는 곱하기와 삼각계산이 들어 있다. 보다 효과적인 원발생알고리듬은 브리센함의 직선알고리듬에서와 같이 결심파라메터의 증분계산에 기초하고 있는데 그것은 간단한 옹근수연산만을 진행한다.

라스터현시장치에 대한 브리센함의 직선알고리듬은 매 표본걸음에서 원둘레에 제일 가까운 화소를 찾아 내기 위한 결심파라메터를 설정하면 원발생에 적응시킬수 있다. 그러나 원의 방정식 3-24는 비선형이기때문에 원경로로부터 화소사이의 거리를 계산하자면 2차뿌리를 계산하여야 한다. 브리센함의 원발생알고리듬은 화소분할거리들의 두제곱을 비교하는 방법으로 이 2차뿌리계산을 피한다.

직접거리비교방법은 두 화소들사이의 중간위치가 원경계의 안쪽 또는 바깥쪽에 놓이는가를 결정하기 위한 검사이다. 이 방법은 다른 원추곡선들에도 아주 쉽게 적용되며 원의 반경이 옹근수인 경우에는 중점방법이 브리센함의 원알고리듬과 꼭 같은 화소위치를 발생시킨다. 또한 임의의 원추부분을 따라 중점검사를 리용하여 화소위치를 정할 때 포함되는 편차는 화소간격의 절반으로 제한된다.

원의 중점알고리듬

라스터직선알고리듬에서와 같이 단위간격으로 표본화하고 매 걸음에서 주어 진 원경로에 제일 가까운 화소위치를 결정한다. 반경 r와 화면중심위치 $(x_c,\ y_c)$가 주어 진 경우에 먼저 알고리듬을 중심이 자리표원점 $(0,\ 0)$에 있는 원경로주위의 화소위치들을 계산하도록 설정할수 있다. 다음에 매개 계산된 위치 $(x,\ y)$를 x에는 x_c를 더하고 y에는 y_c를 더하여 해당한 화면위치에 옮긴다. 첫 4분구의 $x=0$부터 $x=y$까지의 부분구간에서는 곡선의 경사도가 0부터 -1까지 변한다. 따라서 이 8분구의 정의 x방향에서 단위걸음을 취할수 있다. 매 걸음에서 두개의 가능한 위치중 어느것이 원경로에 더 가까운가를

결정하는데 결심파라메터를 리용할수 있다. 다른 7개의 8분구들에서의 위치는 그다음 대칭에 의하여 얻어 진다.

중점방법을 적용하기 위하여 원함수를 다음과 같이 정의한다.

$$f_{\text{circle}}(x, y) = x^2 + y^2 - r^2 \tag{3-27}$$

반경이 r인 원경계우의 임의의 점 (x, y)는 식 $f_{\text{circle}}(x, y) = 0$을 만족시킨다. 점이 원내부에 있으면 원함수는 부이다. 그리고 점이 원밖에 있으면 원함수는 정이다. 개괄하면 임의의 점 (x, y)의 상대적 위치는 원함수의 부호를 검사하여 결정할수 있다.

$$f_{\text{circle}}(x, y) \begin{cases} < 0, & \text{점은원경계안에 있다.} \\ = 0, & \text{점은원경계우에 있다.} \\ > 0, & \text{점은원경계밖에 있다.} \end{cases} \tag{3-28}$$

식 3-28 의 원함수검사는 매 표본걸음에서 원경로에 가까운 화소들사이의 중점에 대하여 진행한다. 그러므로 원함수는 중점 알고리듬에서의 결심파라메터이며 증분계산을 직선알고리듬에서처럼 설정할수 있다.

그림 3-15. 원경로를 따라 표본위치 x_k+1 에서의 후보 화소들사이의 중점

그림 3-15에서는 표본위치 x_k+1에서 두개의 후보화소사이의 중점을 보여 주었다. (x_k, y_k)의 화소가 현시되였다고 하면 다음에 위치 (x_k+1, y_k)의 화소와 위치 (x_k+1, y_k-1)의 화소중 어느것이 원에 더 가까운가를 결정하여야 한다. 결심파라메터는 이 두 화소들사이의 중점에서 평가되는 원함수에 대한 식 3-27이다.

$$\begin{aligned} p_k &= f_{\text{circle}}\left(x_k+1, y_k - \frac{1}{2}\right) \\ &= (x_k+1)^2 + \left(y_k - \frac{1}{2}\right)^2 - r^2 \end{aligned} \tag{3-29}$$

$p_k<0$이면 중점은 원안에 있으며 주사선 y_k의 화소가 원경계에 더 가깝다. 그렇지 않으면 중점은 바깥 또는 원경계우에 있으며 주사선 y_k-1의 화소를 선택한다.

린접한 결심파라메터들은 증분계산을 리용하여 얻는다. 린접한 결심파라메터에 대한 재귀적인 식을 표본위치 $x_{k+1}+1 = x_k+2$ 에서 원함수를 평가하여 얻자.

$$\begin{aligned} p_{k+1} &= f_{\text{circle}}\left(x_{k+1}+1, y_{k+1} - \frac{1}{2}\right) \\ &= [(x_k+1)+1]^2 + \left(y_{k+1} - \frac{1}{2}\right)^2 - r^2 \end{aligned}$$

또는

$$p_{k+1} = p_k + 2(x_k+1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1 \tag{3-30}$$

여기서 y_{k+1}은 p_k의 부호에 따라 y_k이든지 y_{k-1}이다.

p_{k+1}을 얻기 위한 증분은 $2x_{k+1}+1$ (p_k가 부일 때) 혹은 $2x_{k+1}+1-2y_{k+1}$이다. 항 $2x_{k+1}$과 $2y_{k+1}$의 평가도 역시 증분계산으로

$$\begin{aligned} 2x_{k+1} &= 2x_k + 2 \\ 2y_{k+1} &= 2y_k - 2 \end{aligned}$$

와 같이 얻을수 있다. 시작위치 (0, r)에서 이 두 항들은 각각 값 0과 $2r$를 가진다. 매개 린접한 값은 이전의 $2x$값에 2를 더하고 이전의 $2y$의 값에서 2를 덜어 얻는다.

초기결심파라메터는 시작위치 $(x_0, y_0)=(0, r)$에서 원함수를 평가하여 얻는다.

$$p_0 = f_{circle}\left(1, r-\frac{1}{2}\right)$$

$$= 1+\left(r-\frac{1}{2}\right)^2 - r^2$$

또는

$$p_0 = \frac{5}{4} - r \qquad (3-31)$$

만약 반경 r가 옹근수로 주어 지면 간단히 p_0을

$$p_0 = 1 - r \qquad (r\text{는 옹근수})$$

로 둥그리기할수 있으므로 모든 증분은 다 옹근수이다.

브리센함의 직선알고리듬에서와 같이 중점방법은 원파라메터들이 옹근수화면자리표로 주어 진다는 가정하에서 원주위를 따르는 화소위치들을 옹근수더하기와 덜기를 리용하여 계산한다. 원의 중점알고리듬에서의 단계들을 다음과 같이 개괄할수 있다.

원의 중점알고리듬

1. 반경 r와 원중심 (x_c, y_c)를 입력하고 중심이 원점에 있는 원둘레의 첫 점을

   $$(x_0, y_0) = (0, r)$$

   와 같이 얻는다.
2. 결심파라메터의 초기값을 다음과 같이 계산한다.

   $$p_0 = \frac{5}{4} - r$$
3. $k=0$에서 시작하여 매 x_k위치에서 검사를 진행한다.
   $p_k<0$이면 중심이 (0,0)에 있는 원을 따르는 다음점은 (x_k+1, y_k)이며

   $$p_{k+1} = p_k + 2x_{k+1} + 1$$

   그렇지 않으면 원을 따르는 다음점은 (x_k+1, y_k-1)이며

   $$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$$

   여기서 $2x_{k+1} = 2x_k + 2$, $2y_{k+1} = 2y_k - 2$이다.
4. 다른 7개의 8분구들에서의 대칭점을 결정한다.
5. 매 계산된 화소위치 (x, y)를 중심이 (x_c, y_c)에 있는 원경로로 옮기여 자리표값들을 현시한다.

   $$x = x + x_c, \qquad y = y + y_c$$
6. 걸음 3으로부터 5까지를 $x \geq y$일 때까지 반복한다.

실례 3-2. 원의 중점그리기방법

반경 r=10 인 원이 주어 진 경우 x=0 부터 x=y 까지의 첫 4 분구에 있는 8 분구의 원을 따라 점위치들을 결정하는 원의 중점알고리듬을 보자. 결심파라메터의 초기값은

$$p_0 = 1 - r = -9$$

이다.

중심이 자리표원점에 있는 원에 대하여 초기점은 $(x_0, y_0)=(0,10)$이고 결심파라메터계산을 위한 초기증분항들은

$$2x_0 = 0, \qquad 2y_0 = 20$$

이다.

린접한 결심파라메터값들과 원경로를 따르는 위치들은 중점방법을 리용하여 다음과 같이 계산된다.

k	p_k	(x_{k+1}, y_{k+1})	$2x_{k+1}$	$2y_{k+1}$
0	-9	(1, 10)	2	20
1	-6	(2, 10)	4	20
2	-1	(3, 10)	6	20
3	6	(4, 9)	8	18
4	-3	(5, 9)	10	18
5	8	(6, 8)	12	16
6	5	(7, 7)	14	14

첫 4분구에서 얻어 진 화소위치들을 그림 3-16에 보여 주었다.

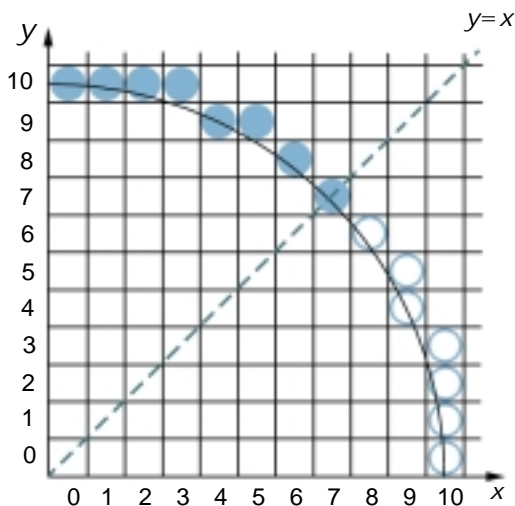



그림 3-16. 원의 중점알고리듬을 리용하여 중심이 원점에 있고 반경 r=10 인 원경로를 따라 선택된 화소위치(색칠한 동그라미), (색칠하지 않은 동그라미는 첫 4 분구에서의 대칭위치를 보여 준다.)

다음의 프로그람은 중점알고리듬을 리용하여 2값현시장치에서 라스터원을 현시한다. 프로그람에 대한 입력자료는 원중심자리표와 반경이다. 원둘레를 따르는 화소위치들에 대한 세기는 setPixel 루틴의 호출에 의하여 프레임완충기배렬에 넣어 진다.

```
#include "device. h"

void circleMidpoint (int xCenter, int yCenter, int radius)
{
  int x = 0;
  int y = radius ;
  int p = 1 - radius;
  void circlePlotPoints (int, int, int, int);

  /* Plot first set of points */
  circlePlotPoints (xCenter, yCenter, x, y) ;

  while (x < y) {
   x++;
   if (P < 0)
     p += 2 * x + 1;
   else {
      y--;
      p += 2 * (x - y) + 1;
   }
   circlePlotPoints (xCenter, yCenter, x, y) ;
 }
}

void circlePlotPoints (int xCenter, int yCenter, int x, int y)
{
  setPixel (xCenter + x, yCenter + y)
  setPixel (xCenter - x, yCenter + y)
  setPixel (xCenter + x, yCenter - y)
  setPixel (xCenter - x, yCenter - y)
  setPixel (xCenter + y, yCenter + x)
  setPixel (xCenter - y, yCenter + x)
  setPixel (xCenter + y, yCenter - x)
  setPixel (xCenter - y, yCenter - x)
}
```

6절. 타원발생알고리듬

타원은 확장된 원이다. 따라서 타원형곡선은 원그리기절차를 주축과 부축을 따라 타원의 서로 다른 크기를 고려하도록 수정하여 얻을수 있다.

타원의 성질

타원은 모든 점들에 대하여 두개의 고정된 위치(초점)로부터의 거리합이 같은 점들의 모임으로

정의된다(그림 3-17). 타원의 임의의 점 $\mathbf{P}=(x,y)$로부터 두 초점까지의 거리를 d_1과 d_2로 현시한다면 타원의 일반방적식은

$$d_1 + d_2 = \text{constant} \tag{3-32}$$

와 같이 표현할수 있다. 거리 d_1, d_2를 초점들의 자리표 $\mathbf{F}_1=(x_1, y_1)$와 $\mathbf{F}_2=(x_2, y_2)$에 의하여 표현하면

$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{(x-x_2)^2+(y-y_2)^2}=\text{constant} \tag{3-33}$$

를 얻는다. 이 방정식을 두제곱하고 남아 있는 2차뿌리를 분리시킨 다음에 다시 두제곱하면 일반타원방정식을

$$Ax^2+By^2+Cxy+Dx+Ey+F=0 \tag{3-34}$$

형으로 다시 쓸수 있다. 여기서 결수 A,B,C,D,E,F는 초점자리표와 타원의 주축과 부축의 길이에 의하여 계산된다. 주축은 타원의 한쪽에서 다른쪽으로 초점들을 지나 늘인 선분이다. 부축은 두 초점사이의 중간위치(타원의 중심)에서 주축을 2등분하는 타원의 보다 짧은 크기를 말한다.

임의로 놓인 타원을 지적하는 대화식방법은 두 초점과 타원경계우에 한 점을 입력하는것이다. 이 3개의 자리표위치들에 의하여 식 3-33에서의 constant를 계산할수 있다. 그다음 식 3-34의 결수들을 계산하여 타원경로를 따라 화소들을 발생시키는데 리용할수 있다.

타원의 방정식은 주축과 부축이 자리표축들에 평행되도록 방향을 맞추어 놓으면 대단히 간단해진다. 그림 3-18에 주축과 부축이 x 및 y축에 평행되게 놓은《표준위치》의 타원을 보여 준다. 이 실례에서 파라메터 r_x는 반주축, 파라메터 r_y는 반부축을 현시한다. 그림 3-18에 보여 준 타원의 방정식은 타원중심자리표와 파라메터 r_x와 r_y에 의하여

$$\left(\frac{x-x_c}{r_x}\right)^2+\left(\frac{y-y_c}{r_y}\right)^2=1 \tag{3-35}$$

와 같이 쓸수 있다. 극자리표 r와 θ를 리용하면 또한 표준위치의 타원을 보조변수방정식으로 다음과 같이 서술할수 있다.

$$\begin{aligned} x &= x_c + r_x\cos\theta \\ y &= y_c + r_y\sin\theta \end{aligned} \tag{3-36}$$

대칭성고찰은 앞으로 계산을 줄이는데 리용될수 있다. 표준위치의 타원은 4분구들사이에서 대칭이지만 원과는 달리 한 4분구의 두 8분구사이에서는 대칭이 아니다. 그러므로 한 4분구에서 타원호를 따르는 화소위치들을 계산하여야 하며 다음에 대칭성에 의하여 남은 3개 4분구에서의 위치들을 얻는다(그림 3-19).

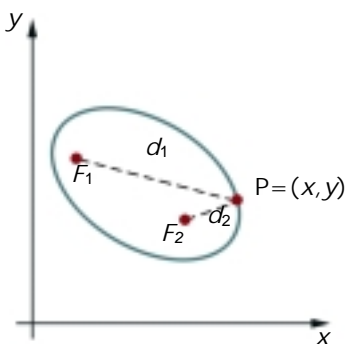



그림 3-17. 초점 $\mathbf{F}_1$과 $\mathbf{F}_2$를 가지는 타원

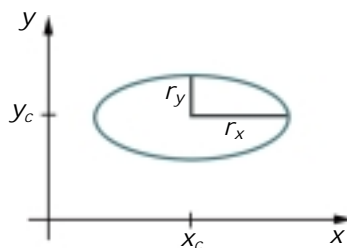



그림 3-18. 반주축 r_x와 반부축 r_y를 가지는 중심이 (x_c, y_c)에 있는 타원

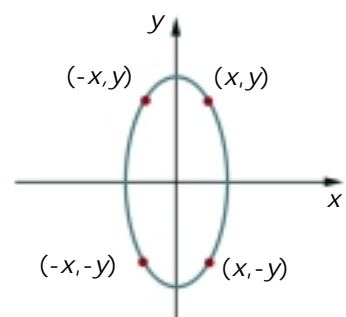



그림 3-19. 타원의 대칭(한 4분구에서 점 (x,y)의 계산은 다른 3개의 4분구에서도 현시할수 있는 타원점을 준다.)

타원의 중점알고리듬

이 방법은 라스터원현시에 리용된것과 류사하다. 파라메터 r_x, r_y 그리고 (x_c, y_c)가 주어 지면 먼저 중심이 원점에 있는 표준위치의 타원에 대한 점 (x, y)들을 결정하고 다음에 타원이 (x_c, y_c)에 중심이 놓이도록 점들을 옮긴다. 또한 비표준위치의 타원을 현시하려면 그다음에 타원을 주축과 부축이 해당한 방향에 놓이도록 중심자리표에 대하여 회전시킨다. 당분간 표준위치의 타원의 현시만을 고찰한다. 물체의 방향과 위치를 변화시키는 일반적인 방법은 5장에서 설명한다.

중점타원방법은 첫 4분구에서 두 부분으로 나뉘여 적용된다. 그림 3-20에서는 $r_x < r_y$인 타원의 경사도에 따라 첫 4분구을 나누는 방법을 보여 주었다. 이 4분구을 곡선의 경사도절대값이 1보다 작은 곳에서는 x방향단위걸음을 취하고 경사도절대값이 1보다 큰 곳에서는 y방향에서 단위걸음을 취하여 처리한다.

구역 1과 2(그림3-20)는 여러가지 방법으로 처리될수 있다. 위치 $(0, r_y)$에서 시작하여 첫 4분구에서 타원경로를 따라 시계바늘방향으로 갈수 있다. 경사도가 -1보다 작아 질 때 x의 단위걸음으로부터 y의 단위걸음에로 넘어 간다. 또 하나는 $(r_x, 0)$에서 시작하여 시계바늘반대방향으로 점들을 선택할수 있으며 경사도가 -1보다 커질 때 y의 단위걸음으로부터 x의 단위걸음으로 바꾼다. 병렬처리기에서는 화소위치들을 두 구역에서 동시에 계산할수 있다. 중점알고리듬의 순차적실현의 실례로서 위치 $(0, r_y)$에서 시작하여 첫 4분구에서 시계바늘방향으로 타원경로를 따라 걸음을 취하도록 한다.

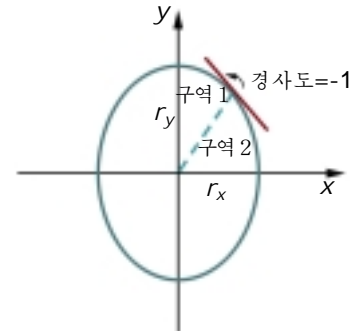



그림 3-20. 타원처리구역(구역 1에서 타원경사도의 절대값은 1보다 작다. 구역 2에서 경사도의 절대값은 1보다 크다.)

타원함수를 식 3-35로부터 $(x_c, y_c)=(0,0)$에 의하여

$$f_{\text{ellipse}}(x,y) = r_y^2x^2 + r_x^2y^2 - r_x^2r_y^2 \qquad (3-37)$$

와 같이 정의한다. 이 함수는 다음의 성질을 가진다.

$$f_{\text{ellipse}}(x,y)\begin{cases} <0, & \text{점이 타원경계안에 있다.} \\ =0, & \text{점이 타원경계우에 있다.} \\ >0, & \text{점이 타원경계밖에 있다.} \end{cases} \qquad (3-38)$$

그러므로 타원함수 $f_{\text{ellipse}}(x,y)$는 중점알고리듬에서 결심파라메터로 쓰인다. 매 표본위치에서 타원경로를 따르는 두개의 후보화소들사이의 중점에서 평가되는 타원함수의 부호에 의해 다음화소를 선택한다.

$(0, r_y)$에서 시작한다면 구역 1과 2사이의 경계에 도달할 때까지는 x방향에서 단위걸음을 취한다(그림 3-20). 다음에 첫 4분구의 남은 곡선우에서 y방향의 단위걸음에로 넘어 간다. 매 걸음에서 곡선의 경사도값을 검사할 필요가 있다. 타원의 경사도는 식 3-37로부터

$$\frac{dy}{dx} = -\frac{2r_y^2x}{2r_x^2y} \qquad (3-39)$$

와 같이 계산된다. 구역 1과 2사이의 경계에서 dy/dx=-1이며

$$2r_y^2x = 2r_x^2y$$

따라서

$$2r_y^2x \geq 2r_x^2y \qquad (3-40)$$

일 때면 구역 1을 벗어 난다.

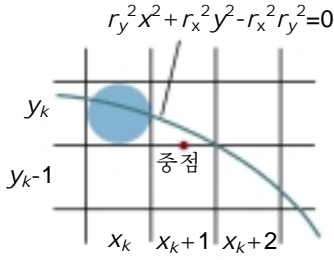



그림 3-21. 타원경로를 따르는 표본위치 x_k +1 에서 후보 화소들사이의 중점

그림 3-21은 첫 구역의 표본위치 x_k+1에서의 두 후보화소들사이의 중점을 보여 준다. 앞걸음에서 위치 (x_k, y_k)가 선택되였다면 타원경로를 따르는 다음위치는 이 중점에서의 결심파라메터(즉 타원함수 3-37)를 평가하여 결정한다.

$$\begin{aligned} p1_k &= f_{\text{ellipse}}\left(x_k+1, y_k-\frac{1}{2}\right) \\ &= r_y^2(x_k+1)^2 + r_x^2\left(y_k-\frac{1}{2}\right)^2 - r_x^2r_y^2 \end{aligned} \tag{3-41}$$

$p1_k$<0이면 중점은 타원안에 있으며 주사선 y_k의 화소가 타원경계에 더 가깝다. 그렇지 않으면 중점은 밖이나 타원경계에 있으므로 주사선 y_k -1의 화소를 선택한다.

구역 1 의 그다음 표본위치 (x_{k+1}+1= x_k +2)에서의 결심파라메터는 다음과 같이 평가된다.

$$\begin{aligned} p1_{k+1} &= f_{\text{ellipse}}\left(x_{k+1}+1, y_{k+1}-\frac{1}{2}\right) \\ &= r_y^2[(x_k+1)+1]^2 + r_x^2\left(y_{k+1}-\frac{1}{2}\right)^2 - r_x^2r_y^2 \end{aligned}$$

또는

$$p1_{k+1} = p1_k + 2r_y^2(x_k+1) + r_y^2 + r_x^2\left[\left(y_{k+1}-\frac{1}{2}\right)^2 - \left(y_k-\frac{1}{2}\right)^2\right] \tag{3-42}$$

여기서 y_{k+1}은 y_k의 부호에 따라 y_k -1 또는 $p1_k$이다.

결심파라메터들은 다음의 크기만큼 증가된다.

$$\text{증분} = \begin{cases} 2r_y^2x_{k+1} + r_y^2, & p1_k < 0\text{인 경우} \\ 2r_y^2x_{k+1} + r_y^2 - 2r_x^2y_{k+1}, & p1_k \ge 0\text{인 경우} \end{cases}$$

원알고리듬에서와 마찬가지로 결심파라메터의 증분은 항 $2r_y^2x$ 와 $2r_x^2y$ 의 값들이 증분으로 얻어지기때문에 더하기 및 덜기만을 리용하여 계산하게 된다. 초기위치 $(0, r_y)$에서 두 항은

$$2r_y^2x = 0 \tag{3-43}$$

$$2r_x^2y = 2r_x^2r_y \tag{3-44}$$

로 평가한다. x와 y가 증가될 때 갱신값들은 식 3-43에 $2r_y^2$을 더하고 식 3-44에서 $2r_x^2$을 덜면 얻어진다. 갱신값들은 매 걸음에서 비교되며 조건 3-40이 만족될 때 구역 1에서 구역 2로 넘어 간다.

구역 1에서 결심파라메터의 초기값은 시작위치 (x_0, y_0)=(0, r_y)에서 타원함수를 평가하면 얻어진다.

$$\begin{aligned} p1_0 &= f_{\text{ellipse}}\left(1, r_y-\frac{1}{2}\right) \\ &= r_y^2 + r_x^2\left(r_y-\frac{1}{2}\right)^2 - r_x^2r_y^2 \end{aligned}$$

또는

$$p1_0 = r_y^2 - r_x^2r_y + \frac{1}{4}r_x^2 \tag{3-45}$$

구역 2에서는 부의 y방향으로 단위걸음표본화하며 중간점은 매 걸음에서 수평화소들사이에서 취해 진다(그림 3-22). 이 구역에서 결심파라메터는

$$\begin{aligned} p2_k &= f_{\text{ellipse}}\left(x_k+\frac{1}{2}, y_k-1\right) \\ &= r_y^2\left(x_k+\frac{1}{2}\right)^2 + r_x^2(y_k-1)^2 - r_x^2 r_y^2 \end{aligned} \quad (3-46)$$

와 같이 평가된다. $p2_k > 0$이면 중점은 타원경계밖에 있으며 x_k에서의 화소를 선택한다. $p2_k \le 0$이면 중점은 안 또는 타원경계에 있으며 화소위치 x_{k+1}을 선택한다.

구역 2에서의 린접한 결심파라메터들사이의 관계를 결정하기 위하여서는 다음표본걸음 $y_{k-1}-1 = y_k-2$에서 타원함수를 평가한다.

$$\begin{aligned} p2_{k+1} &= f_{\text{ellipse}}\left(x_{k+1}+\frac{1}{2}, y_{k+1}-1\right) \\ &= r_y^2\left(x_{k+1}+\frac{1}{2}\right)^2 + r_x^2[(y_k-1)-1]^2 - r_x^2 r_y^2 \end{aligned} \quad (3-47)$$

또는

$$p2_{k+1} = p2_k - 2r_x^2(y_k-1) + r_x^2 + r_y^2\left[\left(x_{k+1}+\frac{1}{2}\right)^2 - \left(x_k+\frac{1}{2}\right)^2\right] \quad (3-48)$$

여기서 x_{k+1}은 $p2_k$의 부호에 따라 x_k 또는 x_k+1로 설정된다.

구역 2에 들어 갈 때 초기위치 (x_0, y_0)은 구역 1에서 선택된 마지막위치로 취한다. 그러면 구역 2에서의 초기결심파라메터는

$$\begin{aligned} p2_0 &= f_{\text{ellipse}}\left(x_0+\frac{1}{2}, y_0-1\right) \\ &= r_y^2\left(x_0+\frac{1}{2}\right)^2 + r_x^2(y_0-1)^2 - r_x^2 r_y^2 \end{aligned} \quad (3-49)$$

$p2_0$의 계산을 간단히 하기 위하여 $(r_x, 0)$에서 시작하여 시계바늘반대방향으로 화소위치들을 선택할수 있다. 그러면 단위걸음은 정의 y방향으로 구역 1에서 선택된 마지막위치까지 취해 지게 된다.

그림 3-22. 타원경계를 따르는 표본위치 y_k-1에서 후보화소들사이의 중점

중점알고리듬은 식 3-34의 타원함수를 리용하여 전체 타원경로에서의 화소위치들을 계산함으로써 비표준위치의 타원을 발생시키는데 적응시킬수 있다. 또 하나의 방법은 5장에서 설명되는 변환방법들을 리용하여 타원축들을 표준위치에 맞추어놓고 중점알고리듬을 적용하여 곡선위치를 결정한 다음에 계산된 화소위치들을 다시 본래의 타원방향에 따르는 경로위치에로 변환할수도 있다.

r_x, r_y 그리고 타원중심이 옹근수화면자리표로 주어 진다면 타원의 중점알고리듬에서 결심파라메터값을 결정하는데 증분의 옹근수계산만 필요하다. 증분 r_x^2, r_y^2, $2r_x^2$, $2r_y^2$들은 절차를 시작할 때 한번만 계산한다. 타원의 중점알고리듬의 개요를 다음걸음들로 적어 놓았다.

타원의 중점알고리듬

1. r_x, r_y 와 타원중심 (x_c, y_c)을 입력하고 중심이 원점에 있는 타원의 첫 점을 얻는다.

$$(x_0, y_0) = (0, r_y)$$

2. 구역 1에서 결심파라메터의 초기값을 계산한다.

$$p1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2 x$$

3. k=0에서 시작하여 구역 1의 매 x_k위치에서 다음의 검사를 진행한다. $p1_k < 0$ 이면 중심이 (0, 0)에 있는 타원을 따르는 다음점은 (x_{k+1}, y_k)이며

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} + r_y^2$$

그렇지 않으면 타원을 따르는 다음점은 (x_k+1, y_k-1)이며

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$$

여기서 $2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2$, $\quad 2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$

그리고 $2r_y^2 x \geq 2r_x^2 y$ 일 때까지 계속한다.

4. 구역 1에서 계산된 마지막점 (x_0, y_0)을 리용하여 구역 2에서 결심파라메터의 초기값을 계산한다.

$$p2_0 = r_y^2 \left(x_0 + \frac{1}{2} \right)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$$

5. k =0 에서 시작하여 구역 2 의 매 y_k위치에서 다음의 검사를 진행한다. $p2_k > 0$ 이면 중심이 (0,0)에 있는 타원을 따르는 다음점은 (x_k, y_k-1)이며

$$p2_{k+1} = p2_k - 2r_x^2 y_{k+1} + r_x^2$$

그렇지 않으면 타원을 따르는 다음 점은 (x_k+1, y_k-1)이며

$$p2_{k+1} = p2_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$$

구역 1에서처럼 x 와 y 에 대하여 같은 증분계산을 리용한다.

6. 다른 3 개의 4 분구에 있는 대칭점들을 결정한다.
7. 매 계산된 화소위치 (x, y)를 중심이 (x_c, y_c)에 있는 타원경로에로 옮겨 자리표값을 현시한다.

$$x = x + x_c, \; y = y + y_c$$

8. $2r_y^2 x \geq 2r_x^2 y$ 일 때까지 구역 1 에 대한 걸음을 반복한다.

실례 3-3. 중점법에 의한 타원그리기

타원파라메터 r_x=8, r_y=6이 입력으로 주어 진 조건에서 첫 4분구에서 타원경로를 따르는 라스터위치를 결정하는것으로써 타원중점알고리듬의 걸음들을 설명한다. 초기값과 결심파라메터계산을 위한 증분은

$$2r_y^2 x = 0 \qquad (\text{증분 } 2r_y^2 = 72\text{로})$$

$$2r_x^2 y = 2r_x^2 r_y \quad (\text{증분 } -2r_x^2 = -128\text{로})$$

구역 1에서 중심이 원점에 있는 타원에 대한 초기점은 $(x_0, y_0)=(0,6)$이며 초기결심파라메터값은

$$p1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2 = -332$$

련접한 결심파라메터값과 타원경로를 따르는 위치들은 중점방법을 리용하여 다음과 같이 계산된다.

k	$p1_k$	(x_{k+1}, y_{k+1})	$2r_y^2 x_{k+1}$	$2r_x^2 y_{k+1}$
0	-332	(1, 6)	72	768
1	-224	(2, 6)	144	768
2	-44	(3, 6)	216	768
3	208	(4, 5)	288	640
4	-108	(5, 5)	360	640
5	288	(6, 4)	432	512
6	244	(7, 3)	504	384

이제는 $2r_y^2 x > 2r_x^2 y$ 이므로 구역 1에서 벗어 난다.

구역 2에서 초기점은 $(x_0, y_0)=(7,3)$이고 초기결심파라메터는

$$p2_0 = f\left(7 + \frac{1}{2}, 2\right) = -151$$

첫 4분구에서 타원경로를 따르는 남은 위치들은 다음과 같이 계산된다.

k	$p1_k$	(x_{k+1}, y_{k+1})	$2r_y^2 x_{k+1}$	$2r_x^2 y_{k+1}$
0	-151	(8, 2)	576	256
1	233	(8, 1)	576	128
2	745	(8, 0)	-	-

첫 4분구안의 타원경계를 따라 선택되는 점위치들의 배치를 그림 3-23에 보여 주었다.

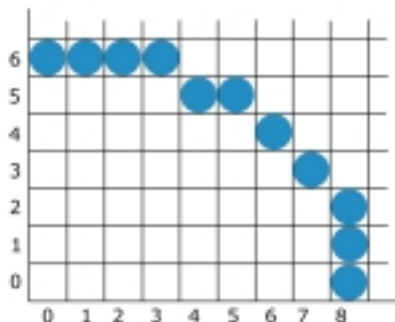



그림 3-23. 중심이 원점에 있고 r_x=8, r_y=6 인 타원경로를 따르는 점들의 위치(첫 4 분공간에서 화소주소를 계산하는데 중점알고리듬을 리용하였다.)

다음의 프로그람의 중점알고리듬은 입력파라메터 Rx, Ry, xCenter, yCenter를 가지는 타원을 현시하는데 리용된다. 첫 4분구에서 곡선을 따르는 위치들이 발생되고 그다음에 자기의 해당한 화면위치

에로 옮긴다. 이 위치와 다른 3개의 4분구에서의 대칭위치들에 대한 세기는 setPixel루틴을 리용하여 프레임완충기에 적재된다.

```
#include "device. h"

#define ROUND(a) ((int)(a+0.5))

void ellipseMidpoint (int xCenter, int yCenter, int Rx, int Ry)
{
  int Rx2 = Rx*Rx;
  int Ry2 = Ry*Ry;
  int twoRx2 = 2*Rx2;
  int twoRy2 = 2*Ry2;
  int p;
  int x = 0;
  int y = Ry;
  int px = 0 ;
  int py = twoRx2 * y;
  void ellipsePlotPoints (int, int, int, int);

  /* Plot the first set of points */
  ellipsePlotPoints (xCenter, yCenter, x, y) ;

  /* Region 1 */
  p = ROUND (Ry2 - (Rx2 * Ry) + (0.25 * Rx2));
  while (px < py) {
   x++ ;
   px += twoRy2 ;
   if (P < 0)
      p += Ry2 + px;
   else {
      Y--;
      py -= twoRx2;
      p += Ry2 + px - py;
   }
    ellipsePlotPoints (xCenter, yCenter, x, y) ;
 }

 /* Region 2 */
 p = ROUND (Ry2*(x+0.5)*(x+0.5) + Rx2*(y-1)*(y-1) - Rx2*Ry2);
 while (y > 0) {
   Y--;
   py -= twoRx2;
   if (P > 0)
      p += Rx2 - py;
   else {
```

```
        x++;
        px += twoRy2 ;
        p += Rx2 - py + px;
      }
      ellipsePlotPoints (xCenter, yCenter, x, y) ;
   }
  }

  void ellipsePlotPoints (int xCenter, int yCenter, int x, int y)
  {
    setPixel (xCenter + x, yCenter + y) ;
    setPixel (xCenter - x, yCenter + y) ;
    setPixel (xCenter + x, yCenter - y) ;
    setPixel (xCenter - x, yCenter - y) ;
  }
```

7절. 기타 곡선

여러가지 곡선함수들은 물체의 모형화, 동화경로지정, 자료와 함수의 그라프그리기, 기타 도형처리응용들에서 쓸모가 있다. 흔히 만나게 되는 곡선들로서는 원추곡선, 삼각 및 지수함수, 확률분포, 일반다항식, 스플라인함수곡선들이 있다. 이 곡선들의 현시는 원 및 타원함수에 대하여 설명된것과 류사한 방법으로 진행할수 있다. 곡선경로를 따르는 위치들은 양함수 $y=f(x)$ 또는 보조변수형식으로부터 직접 얻을수 있다. 다른 하나 음함수 $f(x, y)=0$으로 서술되는 곡선은 중분중점방법을 적용하여 그릴수 있다.

주어 진 곡선함수를 현시하는 간단한 방법은 그것을 선분들로 근사화하는것이다. 이 경우 보조변수표현은 곡선경로를 따라 등간격으로 직선의 끝점위치들을 얻는데 편리하다. 또한 양함수표현으로부터 곡선의 경사도에 따라 독립변수를 선택하는 방법으로 등간격위치들을 발생시킬수도 있다. $y=f(x)$의 경사도절대값이 1보다 작은 곳에서는 x를 독립변수로 선택하고 x를 균일하게 증가시켜 y값을 계산한다. 경사도절대값이 1보다 큰 곳에서는 등간격을 얻기 위하여 역함수 $x=f^{-1}(y)$를 리용하며 등간격 y걸음으로 x의 값을 계산한다.

직선 또는 곡선근사수법은 불련속자리표점들의 자료모임에 대한 그라프를 그리는데 리용된다. 불련속점들은 선분으로 련결할수도 있으며 자료모임을 하나의 직선으로 근사화하기 위하여 선형회귀(최소두제곱법)를 리용할수도 있다. 비선형최소두제곱방법은 자료모임을 어떤 근사함수, 일반적으로는 다항식으로 현시하는데 리용된다.

원 및 타원에서와 같이 적지 않은 함수들은 대칭성을 리용하여 곡선경로를 따르는 자리표위치들의 계산을 줄일수 있다. 실례로 표준확률분포함수는 중심위치(평균)에 대하여 대칭이며 시누스곡선의 한주기사이의 모든 점들은 90°사이의 점들로부터 얻을수 있다.

원추곡선

일반적으로 원추곡선은 2차방정식으로 표현할수 있다.

$$Ax^2 + By^2 + Cxy + Dx + Ey + F = 0 \qquad (3-50)$$

여기서 파라메터 A, B, C, D, E, F 의 값들은 현시되는 곡선의 종류를 결정한다. 결수모임이 주어지면 판별식 $B^2 - 4AC$ 의 평가에 의하여 그려 질 매개 원추곡선을 결정할수 있다.

$$B^2 - 4AC \begin{cases} < 0, & \text{타원(원)} \\ = 0, & \text{포물선} \\ > 0, & \text{쌍곡선} \end{cases} \qquad (3-51)$$

실례로 $A=C=1$, $B=0$, $D=-2x_c, E=-2y_c$, $F = x_c^2 + y_c^2 - r^2$일 때에는 원의 방정식 3-24가 얻어 진다. 식 3-50은 또한 《퇴화》된 원추곡선 즉 점과 직선도 표현한다. 타원, 쌍곡선, 포물선은 일부 동화들에서 아주 잘 리용된다. 이 곡선들은 중력, 전자기력 또는 핵력을 받는 물체들에 대한 궤도나 또는 기타 운동을 표현한다. 실례로 태양계에서 행성의 궤도는 타원이며 균일한 중력마당에 던져 진 물체는 포물선궤도를 따라 움직인다. 그림 3-24에서는 부의 y방향으로 작용하는 중력마당에 대한 표준위치에서의 포물선경로를 보여 주었다. 이 물체의 포물선궤도에 대한 양함수표현은

$$y = y_0 + a(x - x_0)^2 + b(x - x_0) \qquad (3-52)$$

와 같이 쓸수 있다. 여기서 상수 a와 b는 물체의 초기속도 v_0과 균일한 중력에 의한 가속도 g에 의하여 결정된다. 이러한 포물선운동을 초기발사시각으로부터 초단위로 측정되는 시간파라메터 t를 리용하여 보조변수방정식으로도 서술할수 있다.

$$\begin{aligned} x &= x_0 + v_{x0} t \\ y &= y_0 + v_{y0} t - \frac{1}{2} g t^2 \end{aligned} \qquad (3-53)$$

여기서 v_{x0}과 v_{y0}은 초기속도의 성분들이며 지구결면가까이에서의 g의 값은 약 980cm/s^2이다. 이때 포물선경로를 따르는 물체의 위치는 선택된 시간걸음들에서 계산된다.

쌍곡선운동(그림 3-25)은 대전립자들의 충돌 및 어떤 중력문제들과 관련되여 일어 난다. 실례로 태양주위에서 움직이는 혜성 또는 운석은 쌍곡선경로를 따라 움직이며 돌아 오지 않고 바깥공간으로 달아 날수 있다. 물체의 운동을 표현하는 매개 가지(그림 3-25에서 왼쪽 및 오른쪽)는 문제에 포함되는 힘에 관계된다. 그림 3-25에서 중심이 원점에 있는 쌍곡선의 표준방정식을

$$\left(\frac{x}{r_x}\right)^2 - \left(\frac{y}{r_y}\right)^2 = 1 \qquad (3-54)$$

와 같이 쓸수 있다. 여기서 왼쪽 가지에 대해서는 $x \le -r_x$, 오른쪽 가지에 대해서는 $x \ge r_x$이다. 이 방정식은 표준타원방정식 3-35 와 x^2 및 y^2 항사이의 부호에서만 다르므로 쌍곡선경로를 따르는 점들은 타원알고리듬을 약간 수정하여 얻을수 있다.

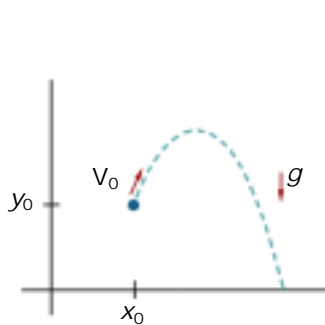



그림 3-24. 초기위치 (x_0, y_0)에서 아래로 향한 중력마당속에서 올려던진 물체의 포물선경로

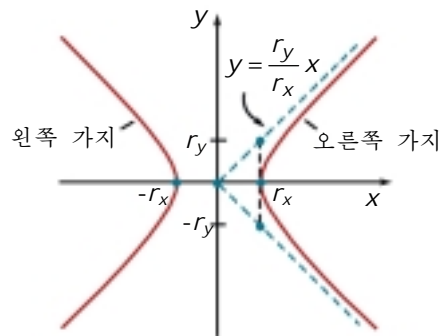



그림 3-25. x축을 대칭축으로 하는 표준위치에서의 쌍곡선의 왼쪽 및 오른쪽 가지

16장에서 동화와 그 방법을 더 자세히 설명한다. 그리고 10장에서 과학적인 가시화에서의 콤퓨터도형처리적응용을 설명한다.

포물선 및 쌍곡선은 대칭축을 가진다. 실례로 식 3-53에 의하여 서술되는 포물선은 축

$$x = x_0 + v_{x0}\, v_{y0}/g$$

에 대하여 대칭이다. 타원의 중점알고리듬에서 리용되는 방법은 두개의 구역((1) 곡선경사도의 절대값이 1보다 작은 구역, (2) 경사도의 절대값이 1보다 큰 구역)에서 쌍곡선 및 포물선경로의 한쪽 대칭축을 따르는 점들을 얻는데 직접 적용할수 있다. 이렇게 하기 위하여서는 먼저 식 3-50에서 해당한 형식을 선택하고 다음에 선택된 함수를 리용하여 두 구역에서 결심파라메터식을 설정한다.

다항식 및 스플라인곡선

x의 n차다항식함수는

$$\begin{aligned} y &= \sum_{k=0}^{n} a_k x^k \\ &= a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} + a_n x^n \end{aligned} \tag{3-55}$$

와 같이 정의된다. 여기서 n은 부아닌 옹근수이며 a_k는 상수, $a_n \neq 0$이다. n=2일 때 2차다항식, n=3일 때 3차다항식, n=4일 때 4차다항식 등을 얻는다. 그리고 n=1일 때는 직선으로 된다. 다항식은 물체형태의 설계, 동화경로의 지정, 자료점들의 불련속모임으로부터 자료경향성그라프그리기를 포함한 많은 도형처리응용에서 쓸모가 있다. 물체의 형태, 또는 운동경로는 흔히 곡선륜곽을 정의하는 몇개의 점들을 지적하고 선택된 점들을 다항식에 맞추는 방법으로 설계한다. 곡선맞추기를 수행하는 한가지 방법은 지적된 점들의 매 쌍사이에 3차다항식곡선부분을 만드는것이다. 이때에 매 부분곡선은 보조변수형식으로

$$x = a_{x0} + a_{x1}u + a_{x2}u^2 + a_{x3}u^3 \tag{3-56}$$

$$y = a_{y0} + a_{y1}u + a_{y2}u^2 + a_{y3}u^3 \tag{3-57}$$

와 같이 서술된다. 여기서 보조변수 u는 구간 [0, 1]에서 변한다. 보조변수방정식에서 u의 결수값들은 부분곡선에서의 경계조건으로부터 결정된다. 한가지 경계조건은 린접한 두 부분곡선이 경계에서 같은 자리표위치를 가진다는것이며 두번째 조건은 하나의 련속적인 원활한 곡선을 얻기 위하여 경계에서 두 곡선의 경사도를 정합시키는것이다(그림 3-26). 다항식토막들에 의하여 형성되는 련속곡선을 **스플라인곡선** 또는 간단히 **스플라인**이라고 한다. 스플라인곡선을 설정하는 다른 방법도 있다. 여러가지 스플라인곡선을 얻는 방법들은 10장에서 고찰한다.

그림 3-26. 지적된 자리표점들사이에서 개별적인 3차다항식부분들로 형성되는 스플라인곡선

8절. 곡선의 병렬알고리듬

곡선발생에서 병렬처리를 리용하는 방법은 선분현시에 리용된것과 류사하다. 곡선구획별로 처리기를 할당하여 순차알고리듬을 적응시킬수도 있고 또 다른 방법을 창안하여 처리기들을 화면구획별로 할당할수도 있다.

원현시를 위한 병렬중점방법은 90°로부터 45°까지의 원호를 크기가 같은 부분호로 나누고 각 처리기를 매 부분호에 할당하는것이다. 병렬브레센함직선알고리듬에서와 같이 매 처리기에 대하여서는 시작 y값과 결심파라메터 p_k값을 결정해야 한다. 화소위치들은 매 부분호별로 계산하며 다른 8분구의 위치들은 대칭에 의하여 얻는다. 류사하게 병렬타원중점방법은 타원호를 첫 4분구에서 등간격부분호로 나누고 이것들을 개별적인 처리기들에 나눈다. 다른 4분구에서의 화소위치들은 대칭성에 의하여 결정한다. 원 및 타원에 대한 화면분할방법은 곡선을 지나가는 매 주사선을 개별적인 처리기에 할당하는것이다. 이 경우 매 처리기는 곡선과의 사귐자리표를 계산하기 위하여 원 또는 타원방정식을 리용하게 된다.

타원호 또는 다른 곡선들의 현시를 위하여서는 간단히 주사선분배방법을 적용할수 있다. 매개 처리기는 사귐위치를 자기에게 할당된 주사선에서 찾기 위하여 곡선방정식을 리용한다. 개별적화소에 할당되는 처리기들이 있는 경우에는 매개 처리기가 곡선으로부터 그에 할당된 화소까지의 거리(또는 거리의 두제곱)를 계산할수 있다. 계산된 거리가 미리 정의된 값보다 작으면 화소로 현시한다.

9절. 곡선함수

대부분의 도형처리프로그람들에는 원, 스플라인 기타 일반적으로 리용되는 곡선들에 대한 루틴이 있다. PHIGS표준은 이 곡선들에 대하여 개별적으로 하나하나 함수를 제공하지 않고 다음과 같은 일반적인 곡선함수를 가지고 있다.

```
generalizedDrawingPrimitive(n, wcPoints, id, datalist)
```

여기서 wcPoints는 매개 자리표위치들의 목록, datalist는 자리표가 아닌 자료값, 파라메터 id는 요구되는 함수를 선택하는 요소이다. 구체적으로 원은 id=1로, 타원은 id=2, …으로 지적될수 있다.

PHIGS함수를 통한 곡선정의실례로서 원(id=1라고 하자.)은 하나의 중심자리표값을 wcPoints에 할당하고 반경값을 detalist에 할당하여 지적할수 있다. 그러면 이 일반화된 그리기기초요소는 원을 발생시키기 위하여 중점방법과 같은 적당한 알고리듬을 참조하게 된다. 대화식입력으로서 원에서는 두개의 자리표점 즉 중심위치와 원둘레우의 한점에 의하여 정의될수 있다. 류사하게 타원의 대화식지적은 세개의 점 즉 두개의 초점과 타원경계우의 한 점(모두 wcPoints에 기억된다.)에 의하여 진행될수 있다. 표준위치의 타원에 대하여서는 wcPoints에 중심자리표만 할당되고 datalist에 r_x 및 r_y에 대한 값이 할당된다. 조종점들에 의하여 정의되는 스플라인은 조종점자리표들을 wcPoints에 할당하여 얻을수 있다.

원 및 타원을 발생시키는 함수들은 흔히 선의 끝점에 대한 파라메터를 지적하면 부분곡선을 그릴수 있는 능력을 가진다. 파라메터목록의 확장은 그림 3-27에서 보여 주는바와 같이 호에 대하여서는 시작점과 끝점들의 중심각을 지적하도록 한다. 원 및 타원호를 가리키는 다른 방법은 호의 시작과 끝자리표위치를 입력하는것이다.

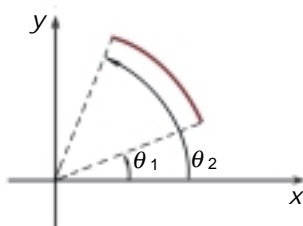



그림 3-27. 시작 및 끝점들의 중심각에 의하여 지적되는 원호(원의 중심은 자리표 원점에 있다.)

10절. 화소주소화 및 물체기하

이제까지는 모든 위치들에 대한 입력을 주사선번호와 주사선에서의 화소위치번호로 주어 진다고 가정하였다. 2장에서 본바와 같이 일반적으로 그림의 지적과 발생에 관계되는 여러가지 자리표계들이 있다. 물체서술은 개별적인 응용에 알맞게 선택된 세계자리표로 주어 지며 입력된 세계자리표는 최종적으로는 화면현시장치의 위치로 변환된다. 물체의 세계자리표서술은 정확한 자리표위치로 주어지는데 그것은 무한히 작은 수학적인 점이다. 그러나 화소자리표는 유한한 화면구역을 대표한다. 세계자리표로 지적되는 물체의 기하학적구조를 보존하자면 수학적인 점이 유한한 크기의 화소구역으로 현시되는것을 보상하여야 한다. 이렇게 하기 위한 한가지 방법은 물체의 경계와 화소구역의 겹침량을 고려하여 간단히 현시되는 물체의 크기를 조종하는것이다. 다른 방법은 물체의 경계가 화소중심이 아니라 화소들의 경계에 놓이도록 세계자리표를 화소들사이의 화면위치에 놓는것이다.

화면살창자리표

현시위치를 화소중심에 의하여 주소화하는것과 다른 한가지 방법은 화면자리표를 단위간격으로 놓여 있는 수평 및 수직화소경계선들의 살창에 의하여 참조하는것이다(그림 3-28). 그러면 화면자리표위치는 두 화소사이의 살창사귐위치를 식별하는 옹근수값쌍으로 된다. 실례로 화면에서 끝점 (0, 0), (5, 2), (1, 4)를 가지는 절선에 대한 수학적인 선경로를 그림 3-29에 보여 주었다.

자리표원점은 화면의 왼쪽 아래에 있으며 매개 화소구역은 그의 왼쪽 아래구석의 옹근수살창자리표로 지적할수 있다. 그림 3-30에서는 8×8라스터부분에 대한 이 변환을 화면자리표위치 (4,5)에 찍혀 진 하나의 화소로 보여 주고 있다. 일반적으로 화면자리표가 (x, y)인 화소가 차지하는 구역은 대각으로 맞은편구석이 (x, y)와 (x+1, y+1)인 단위바른4각형으로 본다. 이 화소주소화방법은 여러가지 우점을 가진다. 그것은 절반짜리 옹근수화소경계를 피하게 하고 정확한 물체표현을 쉽게 하며 많은 주사변환알고리듬과 기타 라스터절차에 포함되는 처리를 간단하게 한다. 앞절에서 설명한 직선긋기 및 곡선발생알고리듬은 화면살창자리표로 표현된 위치들의 입력에 적용될 때에도 여전히 유효하다. 이 알고리듬들에서의 결심파라메터는 화소중심으로부터의 편차의 차이가 아니라 간단히 화면살창편차들의 차이의 크기로 된다.

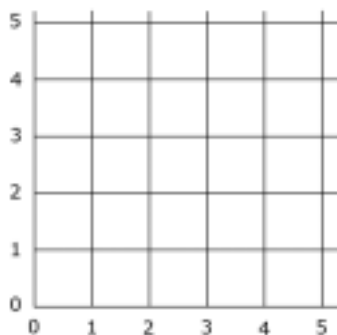



그림 3-28. 옹근수자리표위치를 가지는 화면살창의 왼쪽 아래부분

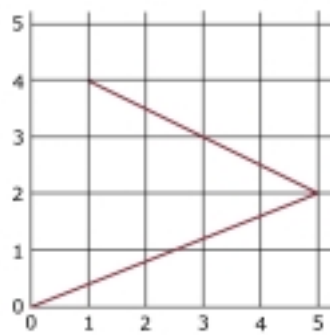



그림 3-29. 화면살창자리표위치사이에서 련결된 선분들에 대한 선경로

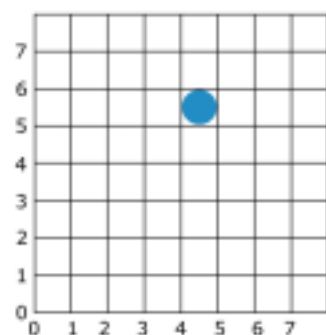



그림 3-30. 라스터위치 (4,5)에 찍혀 진 화소

현시되는 물체의 기하학적인 성질의 유지

물체의 기하학적인 서술이 화소표현으로 변환될 때에는 수학적인 점과 직선이 유한한 화면구역

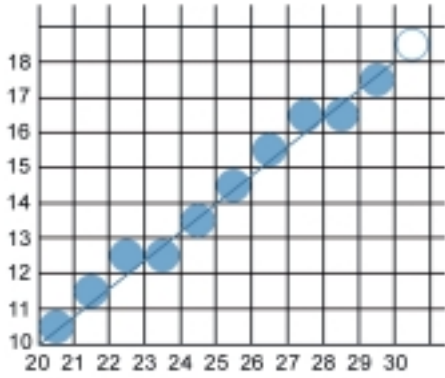



그림 3-31. 입력된 화면살창의 끝점 자리표 (20,10)과 (30,18)에 대한 선경로와 대응하는 화소현시

으로 변환된다. 따라서 만약 입력자리표에 의하여 지적되는 물체의 본래의 기하학적인 크기를 유지하려한다면 물체의 정의를 화면현시로 변환할 때 화소의 유한한 크기를 고려해야 한다. 그림 3-31에서는 2절의 브리센함의 직선알고리듬실례에서 현시되는 직선을 보여 주었다. 직선의 끝점 (20,10)과 (30,18)을 정확한 살창사귐위치로 해석하면 직선은 화면살창위치 (30,18)을 넘어 가서는 안된다는것을 알수 있다. 만약 2절에서 본 실례에서와 같이 화면자리표가 (30,18)인 화소를 현시한다면 11개의 수평단위와 9개의 수직단위를 가지는 직선을 현시하게 된다. 그러나 수학적인 직선에서는 $\Delta x=10$, $\Delta y=8$이다. 만약 화소를 그의 중심위치에 의하여 주소화한다면 현시되는 직선의 길이는 한개 끝점화소를 없애버리는 방법으로 조정할수 있다. 만약 화면자리표를 화소경계의 주소화로 생각한다면 그림 3-31에 보여 준바와 같이 선경로내부의 화소들 즉 선끝점들사이의 화소들만을 리용하여 선을 현시한다. 이 실례에서 제일 왼쪽 화소는 (20,10)에, 제일 오른쪽 화소는 (29,17)에 현시되게 된다. 이것은 (20,10)부터 (30,18)까지의 수학적인 직선과 같은 기하학적크기를 가지는 직선을 현시한다.

경계로 둘러 싸인 구역에 대하여서는 구역을 물체경계내부의 화소들만 가지고 구역을 현시함으로써 입력된 기하학적성질들을 유지할수 있다. 실례로 그림 3-32 ㄱ에서 보여 준 화면자리표정점들로 정의되는 직4각형은 지적된 정점들을 련결하는 테두리화소선까지 포함한 화소들로 채워 현시하면 더 커진다. 원래 정의된 직4각형구역은 12단위이지만 그림 3-32 ㄴ의 현시에서는 20단위의 구역을 가진다. 그림 3-32 ㄷ에서는 본래의 내부화소들만을 현시하므로 직4각형크기가 유지되고 있다. 입력직4각형의 오른쪽 경계는 $x=4$에 있다. 현시에서는 이 경계를 유지하기 위하여 제일 오른쪽 화소의 살창자리표를 $x=3$에 설정한다. 그러면 $x=3$부터 $x=4$까지의 구간은 이 수직렬안의 화소들에 의하여 채워 진다. 류사하게 직4각형의 수학적인 웃경계는 $y=3$에 있으므로 현시되는 직4각형에 대한 웃화소행을 $y=2$에 설정한다.

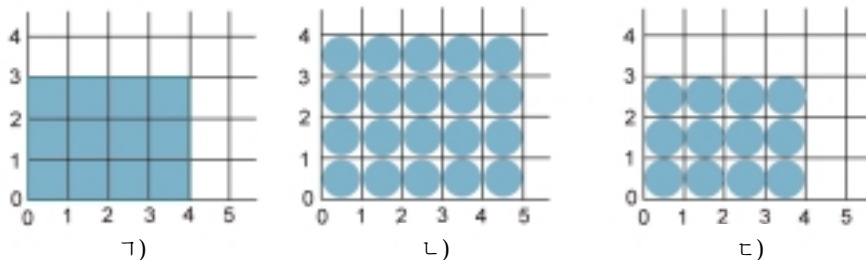



그림 3-32. 화면자리표 (0, 0), (4, 0), (4, 3), (0, 3)인 정점들을 가지는 직 4 각형(ㄱ)의 오른쪽과 웃경계를 포함하는 화면(ㄴ), 기하학적크기를 유지하는 화면(ㄷ)에로의 변환

물체경계의 유한한 화소너비에 대한 이런 보상을 다른 다각형들과 곡선도형들에도 적용함으로써 라스터현시장치에서 입력물체를 지적된 크기로 유지할수 있다. 실례로 반경이 5이고 중심위치가 (10, 10)인 원은 화면살창자리표위치를 리용한 원의 중점알고리듬에 의하여서는 그림 3-33에서와 같이 현시되게 된다. 그런데 그려 진 원의 직경은 11이다. 직경 10으로 주어 진 원을 현시하기 위하여서는

그림 3-34에서와 같이 매 화소주사선과 매 화소렬을 줄이도록 원발생알고리듬을 수정하여야 한다. 이렇게 하기 위한 한가지 방법은 세번째 4분구에서 화면자리표 (10, 5)에서 시작하여 원호를 따라 시계바늘방향으로 점들을 발생시키는것이다. 발생된 매개 점에 대한 기타 7개의 원대칭점들은 주사선을 따라서는 x자리표값을 1감소시키고 화소렬을 따라서는 y자리표값을 1감소시켜 발생시킨다. 이런 방법은 타원을 현시할 때 지적된 비례를 유지하도록 타원알고리듬에도 적용된다.

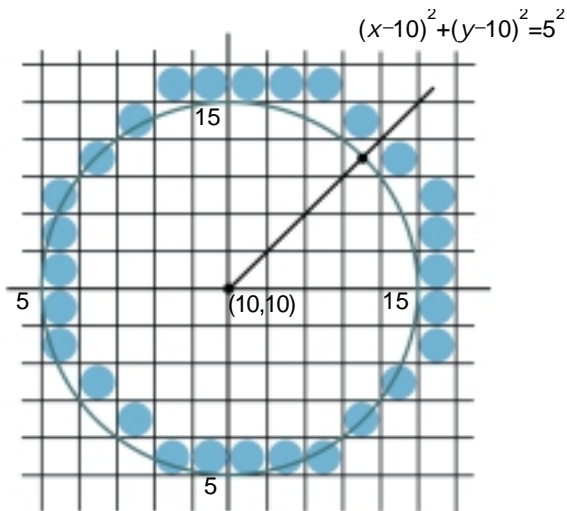



그림 3-33. 화면자리표로 반경이 5인 원의 경로와 중점원 알고리듬으로 그린 원

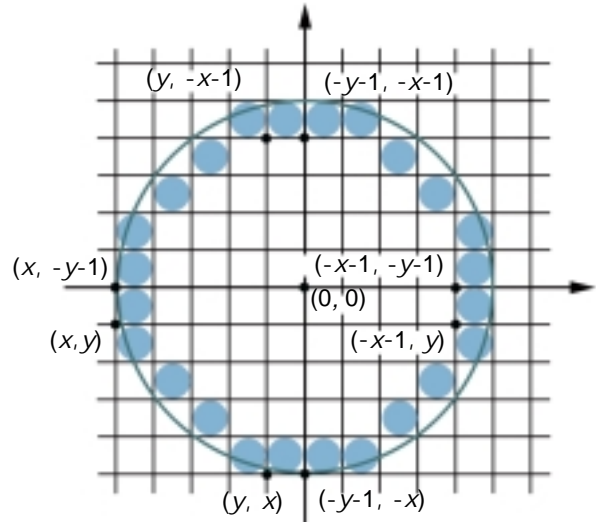



그림 3-34. 지적된 원직경 10을 보존하는 그림 3-33의 원그리기수정

11절. 채운구역기초요소

일반도형처리프로그람들에서 표준출력기초요소의 하나는 색 또는 무늬를 채운 다각형구역이다. 구역기초요소로 때로 다른것들도 사용할수 있으나 다각형은 직선경계를 가지기때문에 처리하기 더 쉽다. 라스터체계의 구역채우기에는 기본적으로 두가지 방법이 있다. 구역을 채우는 한가지 방법은 구역을 지나가는 주사선에 대한 겹침구간을 결정하는것이다. 구역채우기의 다른 방법은 주어 진 내부위치로부터 시작하여 지적된 경계조건을 만날 때까지 이 점으로부터 바깥쪽으로 그려 나가는것이다. 일반도형처리프로그람들에서는 일반적으로 주사선방법을 리용하여 다각형, 원, 타원 기타 간단한 곡선들을 채운다. 내부점으로부터 시작하는 채우기방법은 보다 복잡한 경계와 대화식그리기체계에서 쓸모 있다. 다음부분에서는 지적된 구역의 옹근채우기방법을 고찰한다. 다른 채우기선택항목들은 4장에서 설명한다.

주사선다각형채우기알고리듬

그림 3-35는 다각형구역의 옹근채우기에 대한 주사선절차를 설명한다. 구역채우기알고리듬은 다각형을 지나가는 매 주사선에 대하여 주사선과 다각형변들의 사귐점위치를 결정한다. 이 사귐점들은 왼쪽에서 오른쪽으로 정렬되며 매 사귐점쌍사이의 대응하는 프레임완충기위치들을 지적된 채우기색으로 설정한다. 그림 3-35의 실례에서 다각형경계와 사귀는 4개의 화소위치들은 $x=10$부터 $x=14$까지, $x=18$부터 $x=24$까지 2개의 내부화소구간을 정의한다.

다각형의 정점에서 사귀는 일부 주사선은 특별한 처리를 요구한다. 정점을 통과하는 주사선은 그 위치에서 다각형의 2개의 변과 사귀므로 주사선과의 사귐점목록에 두개의 점으로 넣는다. 그림 3-36

은 변의 끌점들과 사귀는 y 및 y'위치에서의 두개의 주사선을 보여 준다. 주사선 y는 다각형의 5개 변들을 지나간다. 주사선 y에서의 사귐점들은 정확히 내부화소구간들을 식별하고 있다. 그러나 주사선 y'에서 정확한 내부점을 결정하려면 약간한 추가적인 처리를 해야 한다.

그림 3-36에서 주사선 y와 y'사이의 위상학적인 차이는 주사선과 사귀는 변들의 상대적인 위치를 살펴 보면 구별된다. 주사선 y에서는 정점을 공유하면서 사귀는 두 변들이 주사선의 량쪽에 있다. 그렇지만 주사선 y'에서는 사귀는 두개의 변들이 다같이 주사선 웃쪽에 있다. 이리하여 추가적인 처리를 요구하는 정점들은 주사선의 량쪽에서 련결되는 변들을 가지는 점들이다. 이 정점들은 다각형경계를 시계바늘 또는 반대방향순서로 추적하면서 하나의 변으로부터 다른 변으로 움직일 때 정점들의 y자리표에서의 상대적인 변화를 조사하면 구별할수 있다. 만약 련속된 두 변의 끌점들의 y값들이 단조적으로 증가 또는 감소한다면 중간정점은 그것을 통과하는 어떤 주사선에 대하여 하나의 사귐점으로 생각해야 한다. 만약 그렇지 않으면 공유된 정점은 다각형경계에서의 국부극값(최소 또는 최대)을 표현하므로 그 정점을 통과하는 주사선에 대하여 두개 변의 사귐점들로 사귐점목록에 넣어야 한다.

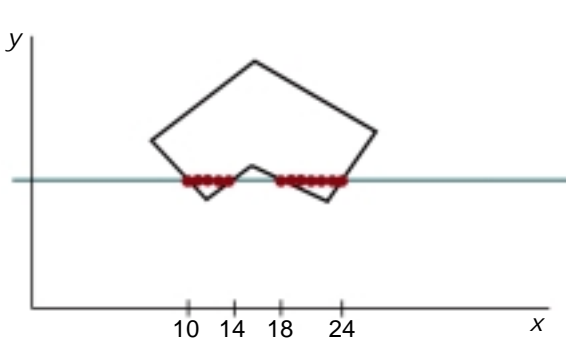



그림 3-35. 다각형구역을 통과하는 주사선에 대한 내부화소들

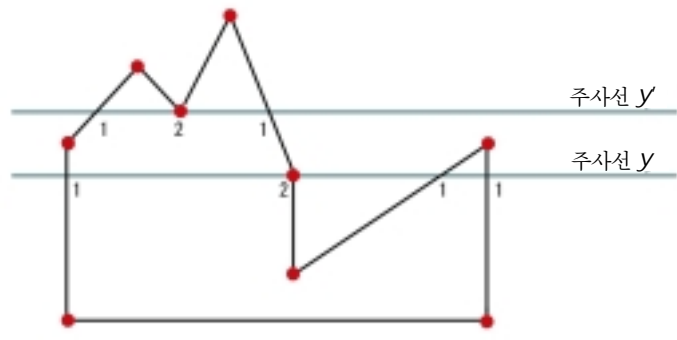



그림 3-36. 다각형의 정점을 지나가는 주사선에 대한 사귐점(주사선 y는 홀수개의 사귐점을 발생시킨다. 그러나 주사선 y'는 내부화소구간들을 정확히 식별할수 있게 짝을 맞출수 있는 짝수개의 사귐점을 발생시킨다.)

정점을 하나의 사귐점으로 생각하는가 또는 두개의 사귐점으로 생각하는가 하는 질문에 대답하는 한가지 방법은 하나의 사귐점으로 생각하여야 할 정점들을 따로 떼놓기 위하여 일부 다각형변들을 짧게 하는것이다. 다각형경계의 비수평변들은 지적된 시계바늘 또는 반대방향순서로 처리할수 있다. 매개 변을 처리할 때 그 변과 다음 비수평변이 단조증가 또는 감소하는 y끌점값을 가지는가를 결정하는 검사를 진행한다. 만약 그렇다면 두 변을 련결하는 공통정점을 통과하는 주사선에 대하여 하나의 사귐점만이 발생된다는것을 담보하기 위하여 아래변을 짧게 한다. 그림 3-37에서는 변을 짧게 하는 방법을 보여 주고 있다. 두 변의 끌점들의 y자리표가 증가할 때는 그림 3-37 ㄱ에서와 같이 현재 변의 웃끌점의 y값을 1감소시킨다. 그림 3-37 ㄴ에서와 같이 끌점들의 y값이 단조감소하고 있을 때는 다음변의 웃끌점의 y자리표를 감소시킨다.

주사변환과 기타 도형처리알고리듬들에서 수행되는 계산은 일반적으로 현시되는 장면의 여러가지 《상관성》(coherence)이라는 유리한 점을 가지고 있다. 여기서 말하는 상관성이란 간단히 장면의 한 부분과 다른 부분사이에는 처리를 줄이는데 리용될수 있는 이러저러한 관계가 있다는 사실을 의미한다. 상관성방법에서는 흔히 하나의 주사선 또는 련속된 주사선들사이에서 증분계산을 적용할수 있다. 변의 사귐점결정에서는 임의의 변의 증분자리표계산을 그 변의 경사도가 한 주사선에서 다음주사선까지에서 일정하다는 사실을 써서 진행할수 있다. 그림 3-38은 다각형의 왼쪽 변을 지나가는 두개의

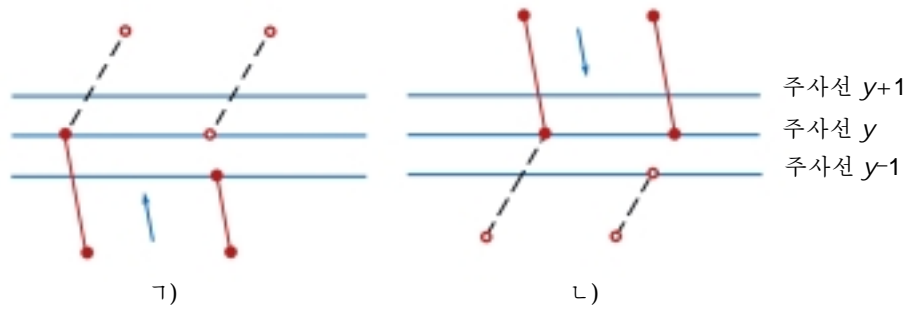



그림 3-37. 다각형둘레를 따라 차례로 변들을 처리할 때 끌점들의 y 값의 조정(현재 처리되고 있는 변은 실선으로 그려 져 있다.)
ㄱ-현재 변의 웃끌점의 y자리표가 1 감소된다,
ㄴ-다음 변의 웃끌점의 y자리표가 1 감소된다.

련속된 주사선을 보여 준다. 이 다각형경계선의 경사도는 주사선과의 사귐점자리표에 의하여 표현할 수 있다.

$$m = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \tag{3-58}$$

두 주사선사이의 y 자리표에서의 변화는 간단히

$$y_{k+1} - y_k = 1 \tag{3-59}$$

이므로 웃주사선에서 x사귐점값 x_{k+1}은 이전의 주사선에서 x사귐점값 x_k로부터

$$x_{k+1} = x_k + \frac{1}{m} \tag{3-60}$$

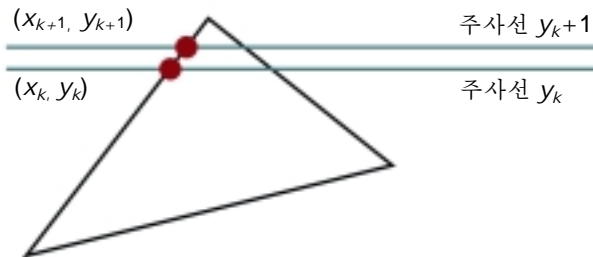



그림 3-38. 다각형경계를 지나가는 두개의 련속된 주사선

와 같이 결정할수 있다. 그러므로 매개 련속한 x자리표는 경사도의 역수를 더하고 제일 가까운 옹근수로 둥그리기하여 계산할수 있다.

채우기알고리듬의 알기 쉬운 병렬적실현은 다각형구역을 지나가는 매 주사선에 개별적인 처리기를 할당하는것이다. 이때에 변의 사귐점계산은 독립적으로 진행한다. 경사도가 m인 변의 초기주사선으로부터 k번째 주사선에 대한 사귐점의 x_k값은

$$x_k = x_0 + \frac{k}{m} \tag{3-61}$$

와 같이 계산할수 있다.

순차적채우기알고리듬에서 변의 x값의 $1/m$만한 증가는 경사도 m이 두 옹근수의 비

$$m = \frac{\Delta y}{\Delta x}$$

라는것을 상기할 때 옹근수연산으로 수행할수 있다. 여기서 Δx와 Δy는 변의 두 끌점들의 x 및 y자리표사이의 차이다. 그러므로 련속된 두 주사선들에 대한 변의 사귐자리표의 증분계산은

$$x_{k+1} = x_k + \frac{\Delta x}{\Delta y} \qquad (3-62)$$

와 같이 표현할수 있다. 이 식을 리용하면 x자리표의 옹근수평가를 계수기를 0으로 초기화하고 다음에 새 주사선에로 움직일 때마다 매번 Δx의 값만큼 계수기를 증가시켜 진행할수 있다. 계수기값이 Δy와 같거나 클 때면 현재의 x사귐점값을 1만큼 증가시키고 계수기는 값 Δy만큼 감소시킨다. 이 절차는 x자리표에 대한 옹근수 및 소수부를 유지하면서 다음 옹근수값에 도달할 때까지 소수부를 증가시키는것과 같은것이다.

옹근수증분의 실례로서 경사도가 m=7/3인 변이 있다고 하자. 초기주사선에서 계수기를 0으로 설정하고 계수기증분은 3으로 설정한다. 이 변을 따라 그다음의 3개 주사선으로 움직일 때 계수기에는 련속적으로 값 3, 6, 9가 할당된다. 초기주사선에서부터 세번째 주사선에서 계수기는 7보다 큰 값을 가진다. 그러므로 x사귐점자리표를 1만큼 증가시키고 계수기는 값 9-7=2로 재설정한다. 이 방법으로 변의 웃끝점에 도달할 때까지 주사선사귐점결정을 계속한다. 이러한 계산은 부의 경사도를 가지는 변에 대한 사귐점을 얻을 때에도 진행된다.

증분을 $\Delta y/2$와 비교하도록 변사귐점알고리듬을 수정하면 옹근수위치를 얻는데서 자르기대신에 가장 가까운 화소의 x사귐점값으로 둥그리기할수 있다. 이것은 매 걸음에서 계수기를 값 $2\Delta x$만큼 증가시키고 Δy와 비교하면 옹근수계산으로 진행할수 있다. 증가가 Δy와 같거나 클 때 x값을 1만큼 증가시키고 계수기는 값 $2\Delta y$만큼 감소시킨다. m=7/3인 웃실례에서 이 변의 초기주사선으로부터 첫 몇개 주사선에 대한 계수기값은 6, 12(-2로 줄여 진다), 4, 10(-4로 줄여 진다), 2, 8(-6으로 줄여 진다), 0, 6, 12(-2로 줄여 진다)이다. x는 이 변에 대한 초기주사선으로부터 2, 4, 6, 9번째 주사선들에서 증가하게 된다. 매개 변에서 요구되는 추가적인 계산은 $2\Delta x = \Delta x + \Delta x$와 $2\Delta y = \Delta y + \Delta y$이다.

다각형채우기를 효과적으로 수행하기 위하여 먼저 주사선을 효과적으로 처리하는데 필요한 모든 정보를 포함하는 변표에 다각형경계변들을 정렬시켜 기억시킨다. 변들에 대한 시계바늘방향 또는 반대방향순서의 처리는 변들을 최소 y값(정확히 주사선위치로)에 따라 정렬시켜 기억시키는 바께트정렬방법을 리용할수 있다. 정렬된 변표에는 비수평변들만이 들어 간다. 변들을 처리할 때 또한 정점사귐문제를 해결하기 위하여 어떤 변들은 짧게 할수 있다. 표의 항목들로서는 매개 주사선에 대하여 변의 최대 y값, x사귐점값(아래 정점에서), 변경사도의 역수가 포함된다. 매개 주사선에서는 변들이 왼쪽에서 오른쪽 순서로 정렬된다. 그림 3-39 에서는 다각형과 그와 관련된 정렬된 변표를 보여 주었다.

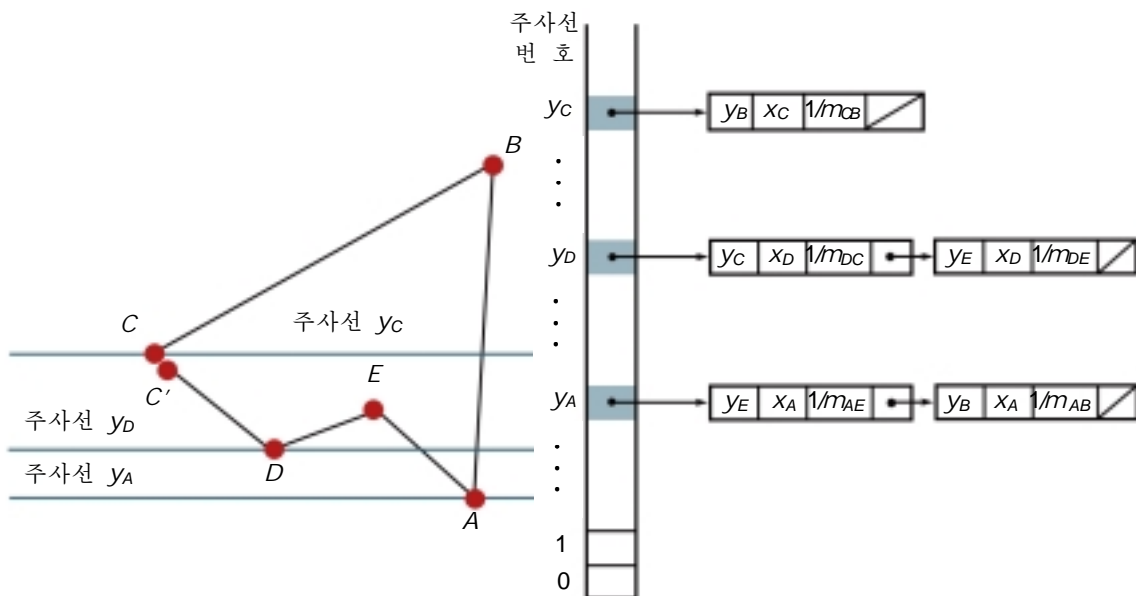



그림 3-39. 다각형과 그의 정렬된 변표. 변 DC는 y 방향에서 1단위 짧아 졌다.

그다음 다각형을 밑에서부터 우로 다각형경계를 지나가는 매 주사선에 대하여 능동변목록을 만들면서 매 주사선들을 처리한다. 주사선에 대한 능동변목록에는 그 주사선이 지나가는 모든 변들이 들어 가며 반복적인 상관성계산으로 변사귐점을 얻는데 리용된다. 변사귐점계산의 실현은 또한 Δx와 Δy값을 정렬된 변표에 기억시켜 놓으면 쉬워 질수 있다. 또한 지적된 다각형내부를 정확히 채우도록 하기 위하여 10절에서 설명된 고찰도 적용할수 있다. 매 주사선에 대하여 제일 왼쪽 x사귐점값에서 시작하여 제일 오른쪽 x사귐점위치에서 끝나는 x사귐점들의 매 쌍에 대한 화소구간을 채운다. 그리고 다각형 매변은 웃끝점에서 y값을 1단위 감소시킨다. 이 조치는 린접한 다각형들의 화소들이 서로 겹치지 않게 해준다.

다음의 프로그람은 다각형정점들의 입력모임에 대한 옹근채우기주사변환을 진행한다. 다각형의 정점범위안의 매 주사선에 대하여 능동변목록이 설정되고 변사귐점이 계산된다. 매 주사선을 따라 내부채우기가 변사귐점들의 련속된 쌍의 사이에서 왼쪽에서 오른쪽으로 가면서 적용된다.

```
#include "device. h"

typedef struct tEdge {
  int yUpper;
  float xIntersect, dxPerScan;
  struct tEdge * next;
} Edge ;

/* Inserts edge into list in order of increasing xIntersect field.*/
void insertEdge (Edge * list, Edge * edge)
{
  Edge * p, * q = list;

  p = q->next;
  while (p != NULL) {
   if (edge->xIntersect < p->xIntersect)
     p = NULL;
   else {
     q = P;
     p = p->next;
    }
 }
 edge->next = q->next;
 q->next = edge;
}

/* For an index, return y-coordinate of next nonhorizontal line*/
int yNext (int k, int cnt, dcPt * pts)
{
  int j ;
```

```
    if ((k+1) > (cnt-1))
      j = 0;
    else
      j = k + 1;
    while (pts[k].y == pts[j].y)
     if ((j+1) > (cnt-1))
       j = 0;
     else
       j++;
    return (pts[j]. y);
  }

  /* Store lower-y coordinate and inverse slope for each edge.Adjust and store
     upper-y coordinate for edges that are the lower member of a monotonically
      increasing or decreasing pair of edges */
  void makeEdgeRec
    (dcPt lower, dcPt upper, int yComp, Edge * edge, Edge * edges[])
  {
    edge->dxPerScan =
      (float) (upper.x - lower.x) / (upper.y - lower.y);
    edge->xIntersect = lower.x;
    if (upper.y < yComp)
     edge->yUpper = upper.y - 1;
    else
     edge->yUpper = upper.y;
    insertEdge (edges[lower.y], edge);
  }

  void buildEdgeList (int cnt, dcPt * pts. Edge * edges[])
  {
    Edge * edge ;
    dcPt v1, v2;
    int i, yPrev = pts[cnt - 2].y;
    v1.x = pts[cnt-1].x; v1.y = pts[cnt-1].y;
   for (i=0; i<cnt; i++) {
     v2 = pts[i] ;
     if (v1.y != v2.y) {        /* nonhorizontal line */
       edge = (Edge *) malloc (sizeof (Edge));
       if (v1.y < v2.y)          /* up-going edge      */
          makeEdgeRec (v1, v2, yNext (i, cnt, pts), edge, edges);
       else                     /* down-going edge    */
          makeEdgeRec (v2, v1, yPrev, edge, edges);
     }
     yPrev = v1 .y;
     v1 = v2;
  }
 }
```

```
void buildActiveList (int scan, Edge * active, Edge * edges[])
{
  Edge * p, * q;

  p = edges[scan]->next ;
  while (p) {
    q = p->next;
    insertEdge (active, p) ;
    P = q;
 }
}

void fillScan (int scan, Edge * active)
{
  Edge * p1, * p2 ;
  int i ;

  p1 = active->next;
  while (p1) {
    p2 = p1->next;
    for (i=p1->xIntersect; i<p2->xIntersect; i++)
      setPixel ((int) i, scan) ;
    p1 = p2->next;
 }
}

void deleteAfter (Edge * q)
{
  Edge * p = q->next;

  q->next = p->next;
  free (p);
}

/* Delete completed edges. Update 'xIntersect' field for others */
void updateActiveList (int scan. Edge * active)
{
  Edge * q = active, * p = active->next;

  while (p)
    if (scan >= p->yUpper) {
       p = p->next;
       deleteAfter (q) ;
    }
    else {
       p->xIntersect = p->xIntersect + p->dxPerScan;
```

```
            q = P;
            p = p->next;
        }
    }

    void resortActiveList (Edge * active)
    {
      Edge * q, * p = active->next;

      active->next = NULL;
      while (p) {
        q = p->next;
        insertEdge (active, p) ;
        P = q;
     }
     }
     void scanFill (int cnt, dcPt * pts)
     {
       Edge * edges[WINDOW_HEIGHT], * active;
       int i, scan;

       for (i=0; i<WINDOW_HEIGHT; i++) {
        edges[i] = (Edge *) malloc (sizeof (Edge));
        edges[i]->next = NULL;
      }
      buildEdgeList (cnt, pts, edges);
      active = (Edge *) malloc (sizeof (Edge));
      active->next = NULL;

      for (scan=0; scan<WINDOW_HEIGHT; scan++) {
        buildActiveList (scan, active, edges);
        if (active->next) {
           fillScan (scan, active);
           updateActiveList (scan, active);
            resortActiveList (active) ;
        }
      }
      /* Free edge records that have been malloc'ed ... */
     }
```

내부외부검사

구역채우기알고리듬과 기타 도형처리과정들에서는 흔히 물체의 내부구역을 식별할 필요가 제기된다. 이제까지는 구역채우기를 표준다각형형태에 의해서만 설명하였다. 초등기하학에서는 다각형을 보통 자체 교차가 없는것으로 정의한다. 표준다각형의 실례로는 3각형, 직4각형, 8각형, 10각형들을 들수 있다. 이런 물체들의 요소변들은 정점들에서만 련결되고 그밖에 평면에서 공통점을 가지지 않

는다. 표준다각형의 내부구역의 식별과정은 일반적으로 간단하다. 그런데 대부분의 도형처리응용들에서는 그림 3-40에서와 같이 교차되는 변을 낳는 순서렬을 비롯하여 채우기구역의 정점들에 대하여 임의의 순서렬이 지적될수 있다. 이러한 형태들에서는 xy평면의 어느 구역을《내부》라고 불러야 하며 어느 구역을 물체의 《외부》라고 지적해야 하는가가 항상 명백한것이 아니다. 도형처리프로그람들에서는 일반적으로 물체의 내부구역을 식별하는데 기우성규칙과 령아닌 감음회수규칙을 리용한다.

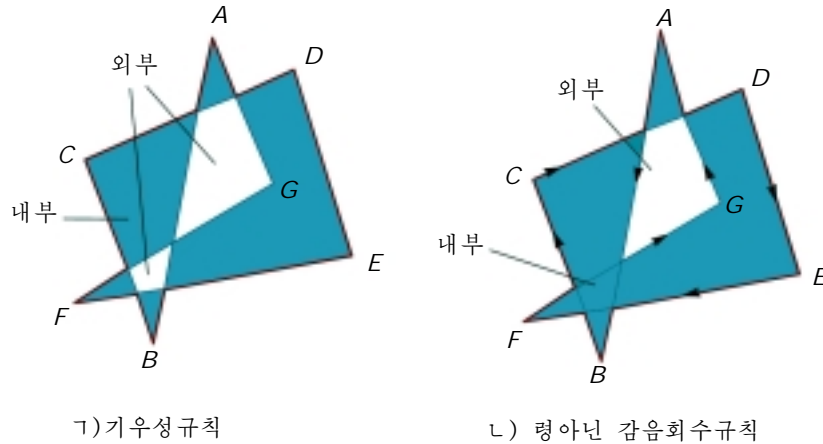



그림 3-40. 자체 교차다각형에 대한 내부 및 외부구역의 식별

기우성규칙(odd-even rule)(**우기성규칙**(even-odd rule)이라고도 부른다.)에서는 임의의 위치 **P**로부터 물체의 자리표범위밖에 놓이는 점에로 가상적으로 직선을 긋고 그것을 지나가는 변의 개수를 센다. 이 직선을 지나가는 다각형변의 개수가 기수이면 **P**는 내부점이다. 그렇지 않으면 **P**는 외부점이다. 변의 개수를 정확히 세기 위하여서는 선택한 직선경로가 다각형의 어느 정점도 지나가지 않게 하여야 한다. 그림 3-40 ㄱ에서는 자체 교차변모임에 대하여 기우성규칙으로부터 얻어 지는 내부 및 외부구역을 보여 주었다. 앞절에서 설명된 주사선다각형채우기알고리듬은 기우성규칙을 리용한 구역채우기의 실례이다.

내부구역을 정의하는 다른 방법은 **령아닌감음회수규칙**(nonzero winding number rule)이다. 이 규칙에서는 개별적인 점에서 나가는 가상적인 어떤 직선에 대하여 다각형의 변들이 시계바늘반대방향으로 감기는 회수를 센다. 이 수를 감음회수라고 부르는데 2차원물체의 내부점은 령아닌 감음회수값을 가지는것으로 정의된다. 령아닌감음회수규칙은 다각형에 대하여 감음회수를 0으로 초기화하고 임의의 위치 **P**로부터 물체의 자리표범위를 벗어 난 점에로 그은 직선을 상상하면서 적용한다. 선택한 직선은 어느 정점도 지나지 말아야 한다. 직선을 따라 위치 **P**로부터 멀리 있는 점에로 움직일 때 매 방향에서 이 직선을 지나가는 변의 개수를 센다. 직선을 오른쪽에서 왼쪽으로 지나가는 다각형변을 만나면 감음회수에 1을 더하고 왼쪽에서 오른쪽으로 지나가는 변을 만나면 1을 던다. 변들의 통과를 다 계수한후의 감음회수의 마지막값이 **P**의 상대적위치를 결정한다. 감음회수가 령이 아니면 **P**는 내부점으로 취하고 그렇지 않으면 **P**는 외부점으로 취해 진다.

그림 3-40 ㄴ에서는 자체 교차변모임에 대한 령아닌감음회수규칙에 의하여 정의한 내부 및 외부구역을 보여 주었다. 표준다각형과 기타 간단한 형태들에서는 령아닌감음회수규칙과 기우성규칙은 같은 결과를 준다. 그러나 보다 복잡한 형태들에서는 두 방법이 그림 3-40의 실례에서와 같이 서로 다른 내부 및 외부구역을 낳을수 있다.

방향변지나가기를 결정하는 한가지 방법은 **P**로부터 멀리 있는 점에로의 직선을 따르는 벡토르 **u**

와 이 직선을 지나가는 매 변의 변벡토르 **E**와의 벡토르적을 취하는것이다. 매개 변에 대한 벡토르적 **u**×**E**의 *z*요소가 정이면 그 변은 오른쪽에서 왼쪽으로 지나가며 감음회수에 1을 더한다. 그렇지 않으면 변은 왼쪽에서 오른쪽으로 지나가며 감음회수에서 1을 던다. 변벡토르는 변의 끌정점위치에서 시작정점위치를 덜어 계산한다. 실례로 그림 3-40의 실레에서 첫 변에 대한 변벡토르는

$$\mathbf{E}_{AB} = \mathbf{V}_B - \mathbf{V}_A$$

이다. 여기서 $\mathbf{V}_A$와 $\mathbf{V}_B$는 정점 A와 B에 대한 점벡토르이다 방향변지나가기를 계산하는 좀 더 간단한 방법은 벡토르적대신에 벡토르의 스칼라적을 리용하는것이다. 이렇게 하기 위하여서는 직선을 따라서 **P**에서 **u**방향으로 내다볼 때 **u**에 수직이면서 오른쪽에서 왼쪽으로 향하는 벡토르를 설정한다. 만약 **u**의 성분이 (u_x, u_y)이면 **u**에 수직인 선은 성분 $(-u_y, u_x)$를 가진다(부록 1). 만약 수직선과 변벡토르의 스칼라적이 정이면 변은 직선을 오른쪽에서 왼쪽으로 지나가며 감음회수에 1을 더한다. 그렇지 않으면 변은 직선을 왼쪽에서 오른쪽으로 가로지르며 감음회수에서 1을 던다.

일부 도형처리묶음들은 령아닌감음회수규칙이 기우성규칙보다 더 다방면적이기때문에 구역채우기를 실현하는데 그것을 리용한다. 일반적으로 물체들은 닫긴곡선의 련결시키지 않은 개별적인 정점들의 모임이나 개별적인 선분들의 모임으로 정의될수 있는데 매개 모임에 대하여 지적되는 방향은 물체의 내부구역을 정의하는데 리용되게 된다. 실례로 자모와 구두점기호와 같은 문자,겹쳐 진 다각형,동심원이나 타원을 들수 있다. 곡선에 대하여서는 기우성규칙을 변의 사귐점을 찾아 내는것이 아니라 곡선경로와의 사귐점을 결정하여 적용한다. 류사하게 령아닌감음회수규칙에서는 위치 **P**에서 나가는 직선과의 사귐점에서 곡선의 접선벡토르를 계산해야 한다.

곡선경계구역의 주사선채우기

일반적으로 곡선경계를 가지는 구역의 주사선채우기에서는 비선형경계를 포함하기때문에 사귐점계산에 다각형채우기보다 더 많은 품이 든다. 원과 타원과 같은 단순한 곡선에 대하여 주사선채우기를 실현하는것은 간단하다. 단지 곡선의 량쪽에서의 2개의 주사선사귐점을 계산하면 된다. 이것은 곡선경계를 따라 화소위치들을 발생시키는것과 같으며 중점방법으로 할수 있다. 그다음 간단히 곡선의 량쪽 경계점들사이의 수평화소구간을 채운다. 4분구의 원에서는 8분구들사이의 대칭을 리용하면 경계계산을 줄인다. 이런 방법은 부분곡선채우기구역을 발생시키는데 리용될수 있다. 실례로 타원호는 그림 3-41에서와 같이 채워 진다. 내부구역은 부분타원호와 호의 시작과 끌점을 련결하여 곡선을 닫는 선분에 의하여 경계 지어 진다. 대칭성과 증분계산을 적용하면 계산을 줄일수 있다.

그림 3-41. 타원호의 내부채우기

경계채우기 알고리듬

구역채우기의 다른 방법은 구역의 내부점에서 시작하여 경계쪽으로 나가면서 내부, 외부를 그리는것이다. 경계가 단일색으로 지적되면 채우기알고리듬은 경계색을 만날 때까지 바깥쪽으로 한 화소씩 나간다. **경계채우기알고리듬**(boundary– fill algorithm)이라고 하는 이 방법은 대화식그림그리기프로그람들에서 내부점이 쉽게 선택되는곳에서 특별히 쓸모가 있다. 미술가 혹은 설계가는 도형입력판 또는 기타 대화식장치를 리용하여 그림의 륜곽선을 먼저 그린 다음 색차림표로부터 채우기색 또는 무늬를 선택하고 내부점을 찍게 된다. 그러면 체계는 도형의 내부를 그린다. 옹근색구역(테두리 없는)을 현시하려면 설계가는 채우기색을 경계색과 같게 선택한다.

경계채우기절차는 입력으로서 내부점자리표 (x, y), 채우기색, 경계색을 받아 들인다. (x, y)에서 시작하여 절차는 린접위치들이 경계색이 아닌가를 결정하기 위한 검사를 진행한다. 아니면 채우기색

으로 그린 다음 다시 그의 린접점들을 검사한다. 이 처리는 구역의 경계색까지의 모든 화소들이 검사될 때까지 계속된다.

구역을 지적하는데는 내부와 외부경계가 다같이 설정될수 있다. 경계채우기를 위한 구역정의의 일부 실례를 그림 3-42에서 보여 주었다.

그림 3-42. 경계채우기절차에 대한 색경계 실례

그림 3-43에서는 현재의 검사위치로부터 린접화소들에로 가는 두가지 방법을 보여 주었다. 그림 3-43 ㄱ에서는 4개 린접점들을 검사한다. 이것은 현재 화소의 오른쪽, 왼쪽, 우, 아래에 있는 화소위치들이다. 이 방법에 의하여 채워진 구역을 **4련결**구역이라고 한다. 그림 3-43 ㄴ에 보여 준 두번째 방법은 보다 복잡한 도형을 채우는데 리용된다. 여기서는 검사될 린접위치모임에 4개의 대각화소들도 포함시킨다. 이 방법을 리용하는 채우기방법을 **8련결**법이라고 한다. 8련결 경계채우기알고리듬은 그림 3-44에서 정의되는 구역의 내부를 정확히 채우지만 4련결경계채우기알고리듬은 부분적으로 채운다.

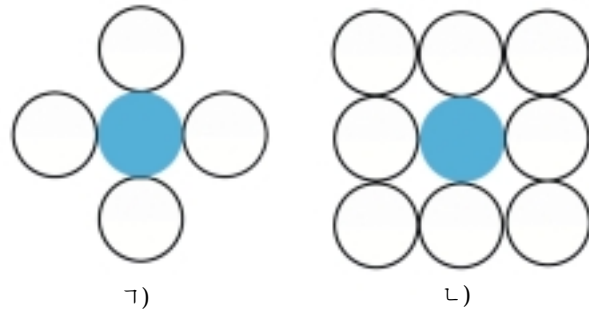



그림 3-43. 4련결구역(ㄱ)과 8련결구역(ㄴ)에 적용되는 채우기방법, 빈동그라미들은 색동그라미의 현재 위치로부터 검사될 화소들을 표현한다.

다음의 프로그람은 파라메터 fill로 지적되는 세기와 파라메터 boundary로 지적되는 경계색을 가지는 4련결구역채우기의 재귀적인 방법을 설명한다. 이 프로그람에 (x+1, y+1)과 같은 대각위치를 검사하는 4개의 추가적인 명령문을 포함시키면 8련결구역채우기로 확장할수 있다.

```
void boundaryFill4 (int x, int y, int fill, int boundary)
{
 int currents;

 current = getPixel (x, y) ;
 if ((current != boundary) && (current != fill)) {
   setColor (fill) ;
   setPixel (x, y) ;
   boundaryFill4 (x+1, y, fill, boundary);
   boundaryFill4 (x-1, y, fill, boundary);
   boundaryFill4 (x, y+1, fill, boundary);
   boundaryFill4 (x, y-1, fill, boundary);
 }
}
```

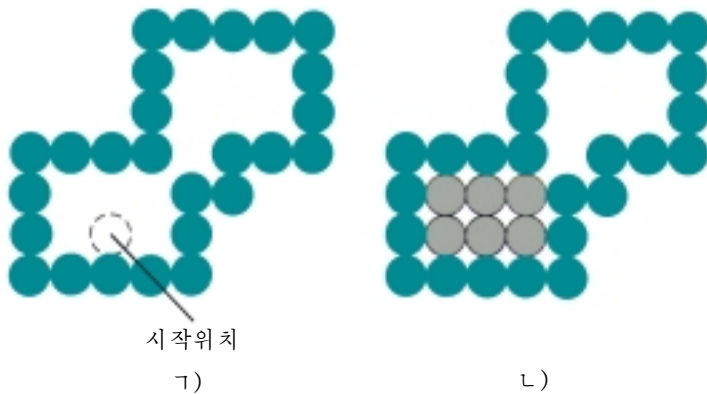



그림 3-44. 색경계안에서 정의되는 구역(ㄱ)은 4련결경계채우기 알고리듬을 리용하면 부분적으로 채워 진다(ㄴ).

재귀적인 경계채우기알고리듬은 일부 내부화소들이 이미 채우기색으로 현시되여 있으면 구역을 정확히 채우지 못할수 있다. 그것은 알고리듬이 다음화소를 경계색과 채우기색에 대하여 다같이 검사하기때문에 일어 난다. 채우기색을 가지는 화소를 만나면 재귀호출을 끝내게 되고 다른 내부화소들을 채우지 않은채로 남겨 놓는다. 이것을 피하기 위하여 경계채우기절차를 적용하기전에 먼저 채우기색으로 초기설정되여 있는 모든 내부화소들의 색을 변화시킨다.

또한 이 절차는 린접점들에 대한 탄창조작이 많이 요구되기때문에 일반적으로 보다 효률적인 다른 방법을 쓴다. 이 방법은 4련결 또는 8련결린접점들을 처리하는것이 아니라 주사선을 따르는 수평화소구간을 채운다. 그러면 현재 위치주위의 처리되지 않은 모든 린접점위치들을 탄창에 넣지 않고 매 수평화소구간에 대한 시작위치만을 탄창에 넣으면 된다. 이 방법으로 초기내부점에서 시작하여 먼저 이 시작주사선우에서 린접된 화소구간들을 채운다. 다음에 린접한 주사선우의 구간들에 대하여 시작위치를 찾아 탄창에 넣는다. 여기서 구간이란 구역경계색으로 현시되는 화소들에 의하여 경계지어 지는 련속된 수평위치렬로 정의한다. 매 다음걸음에서 시작위치를 탄창에서 뽑아 처리를 반복한다.

이 방법을 리용한 화소구간들의 채우기과정을 그림 3-45의 4련결채우기구역을 실례로 설명하자. 이 실례에서는 먼저 시작주사선으로부터 웃경계까지 련속적으로 주사선들을 처리한다. 웃주사선들이 모두 처리된후에는 남아 있는 주사선들에서의 화소구간들을 아래경계로 내려 가면서 채운다. 그림 3-45에 보여 준바와 같이 매개 수평구간의 제일 왼쪽 화소위치가 련속된 주사선들을 따라서 왼쪽에서 오른쪽 순서로 지정되여 탄창에 넣어 진다. 이 그림의 ㄱ에서 초기구간이 채워 지고 그다음의 주사선(아래 및 우)에서의 구간들에 대한 시작위치 1과 2가 탄창에 넣어 진다. 그림 3-45 ㄴ에서 위치 2를 탄창에서 뽑아 처리하여 그림에 보여 준 채워 진 구간을 만든다. 다음주사선의 하나의 구간에 대한 시작화소의 위치 3이 탄창에 넣어 진다. 위치 3이 처리된후의 채워 진 구간과 탄창에 넣어 진 위치를 그림 3-45 ㄷ에 보여 주었다. 그림 3-45 ㄹ는 지적된 구역의 오른쪽 우에 있는 모든 구간들을 처리한후 채워 진 화소들을 보여 준다. 위치 5는 그다음에 처리되며 구간들은 구역의 왼쪽 우에서 채워 진다. 그다음에 위치 4가 아래주사선들에 대한 처리를 계속하기 위하여 선택된다.

침수채우기알고리듬

때때로 경계가 여러가지 색으로 되여 있는 구역을 채우게(다시 색칠하려고) 될수 있다. 그림 3-46 은 여러가지 서로 다른 색구역에 의하여 테두리가 그어 지는 구역을 보여 준다. 이런 구역에 대하여서는 경계색값을 조사하는 대신에 지적된 내부색을 교체하는 방법으로 그릴수 있다. 이 방법을

침수채우기알고리듬(flood-fill algorithm)이라고 한다. 지적된 내부점 (x, y)로부터 시작하여 내부색

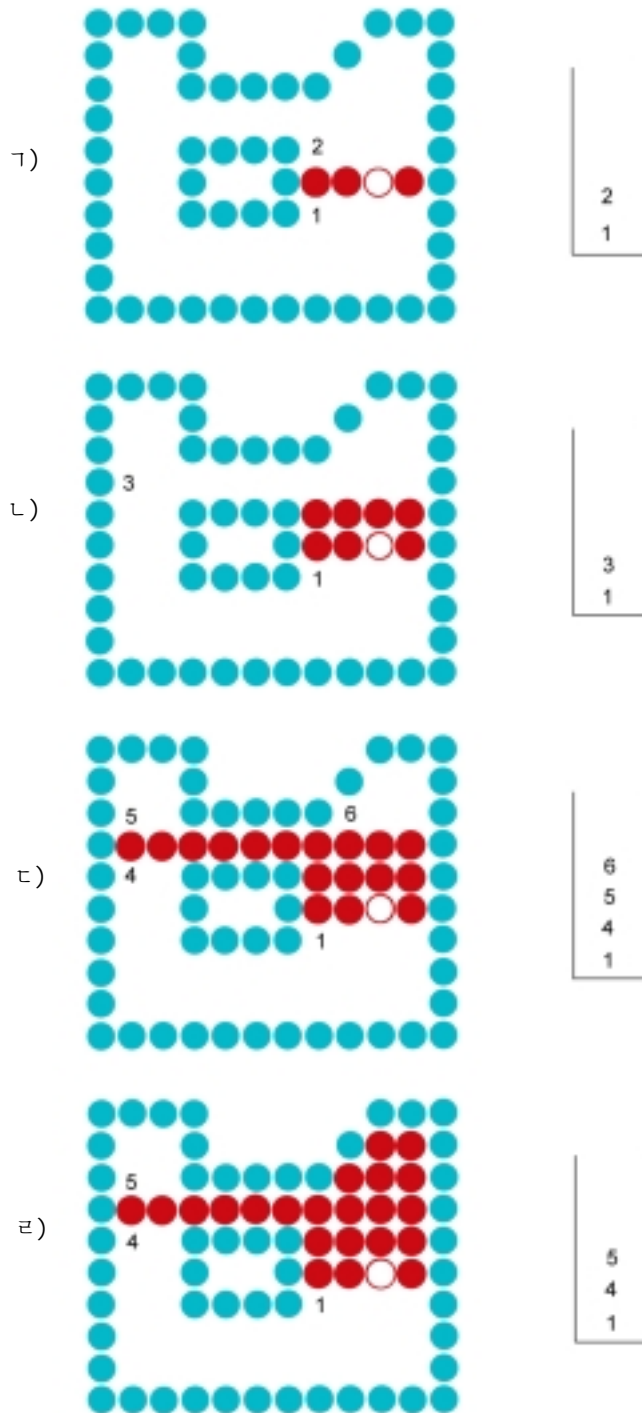



그림 3-45. 4련결구역에 대한 화소구간을 따르는 경계채우기

ㄱ-채워 진 초기화소구간, 초기점(빈동그라미)의 위치와 탄창에 넣어 진 린접한 주사선들에서의 화소구간들의 위치를 보여 준다. ㄴ-초기주사선우의 첫번째 주사선에서 채워 진 화소구간과 탄창의 현재 내용, ㄷ-초기주사선우의 첫 두 주사선들에서의 채워 진 화소구간과 탄창의 현재 내용, ㄹ-정의된 구역의 오른쪽 웃부분에 대한 완성된 화소구간과 탄창에 남아 있는 처리되여야 할 위치

그림 3-46. 다중색경계로 정의되는 구역

으로 현재 설정되여 있는 모든 화소들을 요구하는 채우기색으로 다시 할당한다. 그리려고 하는 구역이 하나이상의 내부색을 가지는 경우에는 먼저 모든 내부점들이 동일한 색을 가지도록 화소값들을 다시 할당할수 있다. 다음에 4련결 또는 8련결방법을 리용하여 모든 내부점들이 다시 그려 질 때까지 화소위치들을 따라 간다. 다음의 프로그람은 입력위치로부터 시작하여 4련결구역을 재귀적으로 침수시켜 채운다.

```
void floodFill4 (int x, int y, int fillColor, int oldColor)
{
 if (getPixel (x, y) == oldColor) {
   setColor (fillColor) ;
   setPixel (x, y) ;
   floodFill4 (x+1, y, fillColor, oldColor);
   floodFill4 (x-1, y, fillColor, oldColor);
   floodFill4 (x, y+1, fillColor, oldColor);
   floodFill4 (x, y-1, fillColor, oldColor);
    }
}
```

경계채우기알고리듬에서 설명된바와 같이 프로그람 floodFill4를 수평화소구간채우기에 의하여 탄창기억기의 요구를 줄이도록 수정할수 있다. 이 방법에서는 oldColor값을 가지는 화소구간들에 대한 시작위치만을 탄창넣기한다. 이 수정된 침수채우기알고리듬의 걸음들은 경계채우기에 대한 그림 3-45에서의 설명과 류사하다. 매개 구간의 첫 위치에서 시작하여 화소값이 oldColor가 아닌 다른 값을 만날 때까지 교체한다.

12절. 구역채우기함수

PHIGS 와 GKS 에서는 채워 진 다각형을 함수

```
fillArea(n, wcVertices)
```

로 현시한다. 현시되는 다각형구역은 wcVertices로 지적되는 정점위치들의 모임을 련결하는 n개의 선분들의 련속으로 경계 지어 진다. 이 프로그람들은 곡선경계를 가지는 물체에 대한 채우기함수들은 제공하지 않는다.

fillArea함수의 실현은 내부채우기방법의 선택에 관계된다. 속이 빈 내부를 둘러 싸는 다각형경계를 현시할수도 있고 다각형현시에서 테두리가 없는 옹근색 또는 문양채우기를 선택할수도 있다. 옹근색채우기에 대한 fillArea함수는 단일색구역을 현시하는 주사선채우기알고리듬으로 실현한다. 채워진 다각형구역의 현시를 위한 PHIGS의 여러가지 속성항목들은 다음장에서 설명한다.

PHIGS에서 사용가능한 다각형의 다른 기초요소는 fillAreaSet이다. 이 함수는 여러개의 다각형에 대하여 매개 다각형의 정점목록을 지적하면 현시될수 있게 한다. 또한 다른 도형처리프로그람들에서는 채워 진 구역현시를 위하여 일반다각형외에 흔히 쓰는 여러가지 함수들도 제공하고 있다. 몇가지 실례를 들면 fillRectangle, fillCircle, fillCircleArc, fillEllipse 그리고 fillEllipseArc이다.

13절. 세포배렬

세포배렬(cell array)은 사용자가 2차원살창무늬로 정의되는 임의의 형태를 현시하게 하는 기초요소이다. 이 함수에 의하여 미리 정의된 색값행렬이 지적된 직4각형자리표구역에 넘겨 진다. 이 함수의 PHIGS 형식은

```
cellArray (wcPoints, n, m, colorArray)
```

이다. 여기서 colorArray는 n×m의 옹근수색값행렬이며 wcPoints는 직4각형자리표구역의 한계 (x_{min}, y_{min})와 (x_{max}, y_{max})를 등록한다. 그림 3-47은 직4각형자리표우에서의 색행렬요소분포를 보여 준다.

그림 3-47에서 매개 자리표세포는 너비 $(x_{max}-x_{min})/m$과 높이 $(y_{max}-y_{min})/n$을 가진다. 화소색값은 화소중심자리표의 상대위치에 따라 할당한다. 화소중심이 $n \times m$의 자리표배렬안에 있으면 그 화소에는 행렬 colorArray의 대응하는 요소의 색이 할당된다.

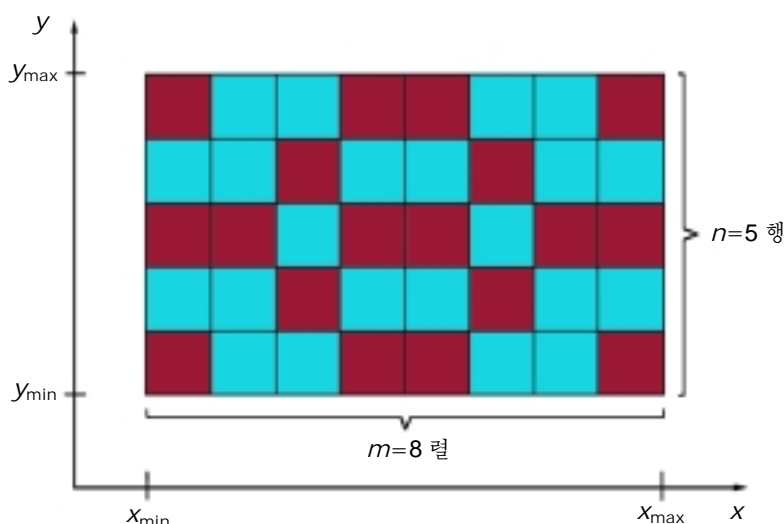



그림 3-47. $n \times m$의 세포행렬의 직4각형자리표구역에로의 넘기기

14절. 문자발생

글자, 수자 기타 문자들은 여러가지 크기와 형태로 현시될수 있다. 문자모임(또는 계렬)에 대한 총체적인 설계모양을 **서체**(type face)라고 한다. 오늘 콤퓨터응용에서 사용할수 있는 서체는 수백개나 된다. 대표적인 몇가지 서체의 실례로는 Courier, Helvetica, New York, Palatino, Zapf Chancery를 들수 있다. 본래 **폰트**(font)란 용어는 10 point Courier Italic 또는 12point Palation Bold와 같이 특정한 크기와 형태로 만들어 진 금속주조활자들의 모임을 말하였다. 지금은 인쇄가 더는 금속주조활자형식으로 진행되지 않기때문에 용어폰트와 서체가 흔히 서로 엇갈려 쓰인다.

서체(또는 폰트)는 두개의 큰 부류 즉 장식이 있는 형과 장식이 없는 형으로 나눌수 있다. 장식형은 주문자획의 끝에서 작은 선 또는 력점을 가지고 있는것이고 무장식형은 이런 것들을 가지지 않은것이다 실례로《Serif type》라는 단어는 장식폰트(serif Font)로 씌여 져 있다. 그러나《Sans-serif type》라는 단어는 무장식폰트로 인쇄되여 있다. 장식형은 일반적으로 읽기 쉽다. 즉 그것은 긴 블로크의 본문을 읽기 쉽다. 한편 무장식형의 개별문자들은 식별하기가 더 쉽다. 이 리유로 무장식형은 보다 알아 보기 쉽다고 말한다. 무장식형문자는 빨리 알아 볼수 있기때문에 이 서체는 표식을 붙이는데나 짧은 머리글부분에 쓰면 좋다.

콤퓨터에 폰트를 기억시키는데는 두가지 서로 다른 표현이 리용된다. 문자모양을 어떤 서체로 표현하는 간단한 방법은 직4각형살창문양을 리용하는것이다. 이때의 문자들의 모임을 **비트배렬폰트**(bitmap font)(또는 **비트배렬된 폰트**)라고 한다. 보다 적응성 있는 다른 방법은 문자모양을 실례로 postScript에서와 같이 직선과 곡선토막들을 리용하여 서술하는것이다. 이 경우의 문자들의 모임을 **륜곽선폰트**(outline font)라고 한다. 그림 3-48에서는 문자표현의 두가지 방법을 보여 주고 있다. 그림 3-48 ㄱ의 문양이 프레임완충기구역에 복사될 때 1로 된 비트는 현시장치에서 현시되여야 할 화소위치를 가리킨다. 그림 3-48 ㄴ의 문자모양을 현시하려면 주사선채우기절차(3장 11절)를 리용하여 문자의 륜곽선의 내부를 채워야 한다.

비트배렬폰트는 정의와 현시가 제일 간단하다. 즉 문자살창을 프레임완충기위치에 넘겨 주기만 하면 된다. 그러나 일반적으로 비트배렬폰트는 매개 변체(크기 및 형태)에 대하여 다 폰트고속기억기에 기억시켜야 하기때문에 더 많은 기억공간이 요구된다. 하나의 모임으로부터 돋움체 및 비낌체와 같은 서로 다른 크기와 변체를 발생시키는것이 가능은 하지만 그것은 보통 좋은 결과를 내지 못한다.

비트배렬폰트와 달리 륜곽선폰트는 매개 변체가 폰트고속기억기를 개별적으로 요구하지 않기때문에 보다 적은 기억기를 요구한다. 문자륜곽선에 대한 곡선정의를 처리하면 돋움체, 비낌체 또는 다른 크기들을 만들수 있다. 그러나 륜곽선폰트는 프레임완충기에 주사변환되여야 하기때문에 그것을 처리하는데 보다 많은 시간이 걸린다.

PHIGS에서는 문자렬을 다음의 함수에 의하여 현시한다.

```
text (wcPoint, string)
```

파라메터 string에 할당된 문자렬이 자리표점 wcPoint에 현시된다. 실례로 명령문

```
text (wcPoint, "Population Distribution")
```

은 wcPoint의 자리표지적으로 인구분포그라프의 표식으로 리용할수 있다. 문자렬이 자리표 (x, y)와 어떻게 상대적으로 위치를 맞추는가 하는것은 사용자의 선택항목이다. 체계는 (x, y)를 현시될 수평문자렬의 첫 문자의 왼쪽 아래구석에 대한 자리표위치로 미리 설정하고 있다. 수직, 수평, 경사와 같은 기타 문자렬쓰기방향은 속성선택항목으로 설정되는데 다음장에서 설명하게 된다.

1	1	1	1	1	1	0	0
0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	0
0	1	1	1	1	1	0	0
0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	0
1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0

ㄱ)

ㄴ)

그림 3-48. 8×8 2값비트배렬문양(ㄱ)과 직선과 곡선토막으로 정의되는 륜곽선모양(ㄴ)으로 표현되는 글자 B

PHIGS의 다른 편리한 문자함수는 **표식기호**(marker symbol)라고 부르는 지시문자를 하나 또는 그 이상의 지적된 위치들에 놓는것이다. 이 함수는 직선함수에서와 같은 파라메터목록으로 정의된다.

```
polymarker (n, wcPoints)
```

이때 미리 정의된 문자가 목록 wcPoints안의 n개 자리표위치의 매개에 중심이 맞추어 진다. polymarker에 의하여 현시되는 미리 설정된 기호는 개별적인 실현에 관계되지만 여기서는 별표(*)가 리용된다고 가정한다.

그림 3-49에는 명령문

```
polymarker (6, wcPoints)
```

에 의한 자료모임의 현시를 보여 준다.

x	y
41	94
59	43
85	74
110	59
121	89
149	122

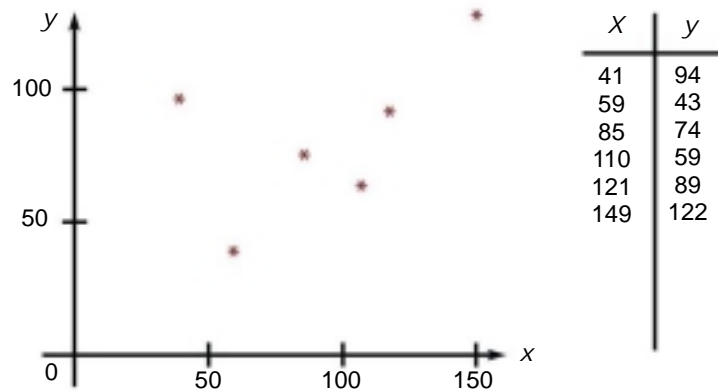



그림 3-49. polymarker 함수에 의하여 현시되는 자료렬

요약

출력기초요소들에 대하여 설명한 이 장에서는 직선, 곡선, 채운구역, 세포배렬무늬와 본문들을 가지고 그림을 만들기 위한 기본적인 도구들을 제공하여 주고 있다.

이러한 기초요소들로써 발생시킨 그림들의 실례를 그림 3-50과 그림 3-51에 보여 주었다. DDA 알고리듬, 브리센함알고리듬, 중점방법은 직선경로를 따라 화소위치들을 현시하는데 사용될수 있는 세가지 방법이다. 직선에서 브리센함알고리듬과 중점법은 동일한것이며 가장 효과적인 방법들이다. 이 방법들에서 프레임완충기호출은 기억기주소계산에서 증분을 써서 효과적으로 진행할수 있다. 직선발생의 모든 알고리듬들은 선분을 분할하여 병렬적으로 실현할수 있다.

원과 타원은 곡선의 대칭성을 고려하면서 중점법을 쓰면 효과적으로 그리고 정확하게 주사변환할수 있다. 기타 원추곡선 즉 포물선,쌍곡선들도 이런 방법으로 그릴수 있다. 하나하나 련속된 다항식들인 스플라인곡선은 설계에서 널리 쓰인다. 곡선경로를 분할하는 방법으로 곡선발생을 병렬적으로 실현할수 있다. 현시되는 직선과 곡선들은 유한한 너비를 가진다는것을 고려하여 물체의 화소치수를 지적된 기하학적치수에 꼭 맞게 조정해야 한다. 이것은 화소위치를 그의 왼쪽 아래모서리로 참조하게끔 주소화하거나 선의 길이를 조정하는 방법으로 진행할수 있다.

채운구역기초요소들은 많은 도형처리프로그람들에서 채운다각형으로 참조한다. 라스터체계에서 다각형채우기를 진행하는 일반적인 방법은 주사선채우기알고리듬이다. 여기서는 다각형을 지나가는 주사선에서의 내부화소구간을 결정한다. 주사선알고리듬은 곡선경계를 가지는 물체의 내부를 채우는데 리용할수 있다. 물체의 내부구역을 채우는 다른 두가지 방법은 경계채우기알고리듬과 침수채우기알고리듬이다.

이 두가지 채우기절차들은 지적된 내부점으로부터 바깥쪽을 향하여 한번에 한점씩 내부를 칠한다. 주사선채우기알고리듬은 내부구역을 가리키기 위하여 기우성규칙을 리용하여 물체의 내부를 채우는 한가지 실례이다.

물체의 내부를 결정하는 다른 방법들도 역시 효과적인데 특히 자체교차가 있는 물체에서 쓸모가 있다. 대표적인 실례가 령아닌 감음회수규칙이다. 이 규칙은 여러가지 경계로 정의되는 물체를 처리하는데서 기우성규칙보다 더 융통성이 있다.

도형처리프로그람들에서 쓸수 있는 보충적인 기초요소들로는 세포배렬, 문자렬, 표식기호가 있다.

세포배렬은 색무늬를 정의하고 기억시키는데 쓴다. 문자렬은 그림과 그라프의 설명을 주는데 쓴다. 그리고 표식기호는 자료점들의 위치를 현시하는데 쓸모가 있다.

표 3-1에 이 장에서 설명된 몇가지 출력기초요소함수들을 주었다.

표 3-1. 실현된 출력기초요소함수

```
typedef struct {float x,y;} wcpt2;
```
2차원세계자리표로 위치를 정의한다.

```
pPolyline(Int n, wcpt2 * pts)
```
pts로 지적된 n-1개의 련결된 선토막들을 그린다.

```
pCircle (wcpt2 center,float r)
```
점 center 에서 반경이 r 인 원을 그린다.

```
pFillarea(Int n, wcpt2 * pts)
```
pts로 지적된 n개 정점을 지나는 채워 진 다각형을 그린다.

```
pCellArray (wcpt2 * pts,Int n,Int m,Int colors)
```
n×m의 colors배렬을 pts로 정의되는 직4각형구역에 넘긴다.

```
pText (wcpt2 position,char * txt)
```
position 위치에 문자렬 txt 를 그린다.

```
pPolymarker (int n, wcpt2 * pts)
```
pts위치들에 n개의 표식기호를 찍는다.

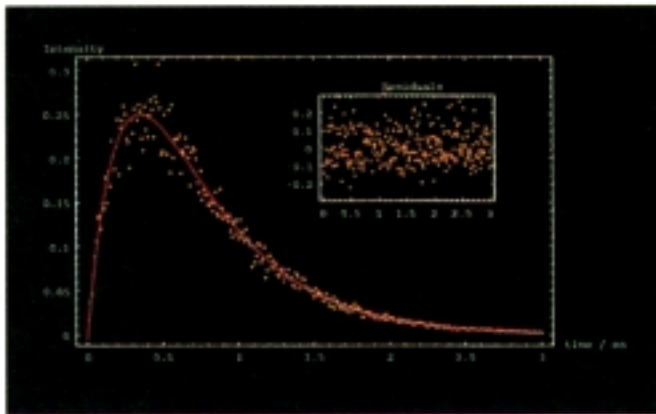



그림 3-50. 선분, 곡선, 동그라미(즉 표식) 그리고 본문으로 만들어 진 자료의 현시

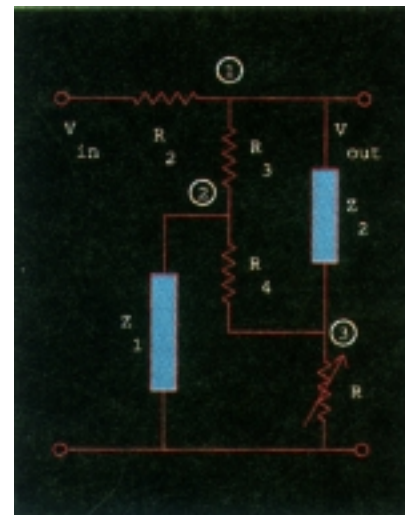



그림 3-51. 선분, 동그라미, 채워 진 직4각형 그리고 본문을 가지고 그린 전자회로도

응용

여기에 출력기초요소들의 응용을 보여 주는 몇가지 실례프로그람들을 제시한다. 표 3-1에 보여 준 함수들과 openGraphics, closeGraphics, setColor, setBackground루틴들이 머리부파일 graphics.h에 정의되여 있다.

첫번째 프로그람은 1년기간의 월별 자료에 대한 선그라프를 만든다. 이 프로그람의 출력이 그림 3-52에 그려 져 있다.

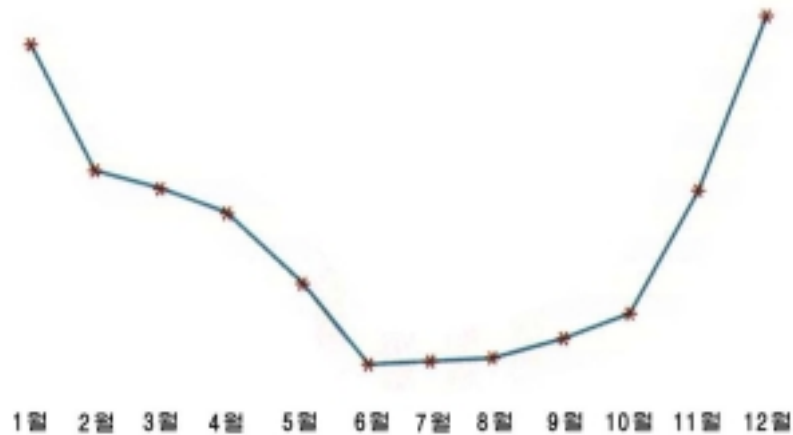



그림 3-52. lineChart절차로 출력한 자료점들의 선현시

이 자료모임은 그림 3-53의 기둥도표를 만들어 내는 두번째 프로그람에서도 리용된다.

```
#include <stdio. h>
#include "graphics. h"

#define WINDOW_WIDTH 600
#define WINDOW_HEIGHT 500
/* Amount of space to leave on each side of the chart */
#define MARGIN_WIDTH 0.05 * WINDOW_WIDTH
#define N_DATA 12

typedef enum
{Jan, Feb. Mar, Apr, May, Jun, Jul, Aug. Sep, Oct, Nov, Dec} Months;

char * monthNames[N_DATA] = {"Jan", "Feb", "Mar", "Apr",
    "May","Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};

int readData (char * inFile, float * data)
{
 int fileError = FALSE;
 FILE * fp;
 Months month;

 if ((fp = fopen (inFile, "r")) == NULL)
   fileError = TRUE;
 else {
   for (month = Jan; month <= Dec; month++)
     fscanf (fp, "%f", &data[month]);
   fclose ( fp) ;
 }
 return (fileError) ;
}
```

```
void lineChart (float * data)
{
 wcPt2 dataPos[N_DATA], labelPos;
 Months m;
 float mWidth = (WINDOW_WIDTH - 2 * MARGIN_WIDTH) / N_DATA;
 int chartBottom = 0.1 * WINDOW_HEIGHT;
 int offset = 0.05 * WINDOW_HEIGHT; /*Space between data and labels*/
 int labelLength =24; /* Assuming fixed-width 8-pixel characters */

 labelPos.y = chartBottom;
 for (m = Jan; m <= Dec; m++) {
    /* Calculate x and y positions for data markers */
   dataPos[m].x = MARGIN_WIDTH + m * mWidth + 0.5 * mWidth;
   dataPos[m].y = chartBottom + offset + data[m];
    /* Shift the label to the left by one-half its length */
   labelPos.x = dataPos[m].x - 0.5 * labelLength;
   pText (labelPos, monthNames[m]);
 }
 pPolyline (N_DATA, dataPos) ;
 pPolymarker (N_DATA, dataPos);
}

void main (int argc, char ** argv)
{
 float data[N_DATA];
 int dataError = FALSE;
 long windowID;

 if (argc < 2) {
   fprintf (stderr, "Usage: %s dataFileName\ n", argv[0]);
   exit () ;
 }
 dataError = readData (argv[I], data);
 if (dataError) {
   fprintf (stderr, "%s error. Can't read file %s\ n", argv[l]);
   exit () ;
 }
 windowID= openGraphics(*argv,WINDOW_WIDTH,WINDOW_HEIGHT);
 setBackground (WHITE) ;
 setColor (BLACK) ;
 lineChart (data) ;
 sleep (10);
 closeGraphics (windowID) ;
}
```

```
void barChart (float * data)
{
  wcPt2 dataPos[4J, labelPos;
  Months m;
  float x, mWidth = (WINDOW_WIDTH - 2 * MARGIN_WIDTH) / N_DATA;
  int chartBottom = 0.1 * WINDOW_HEIGHT;
  int offset = 0. 05 * WINDOW_HEIGHT; /*Space between data and labels*/
  int labelLength =24; /* Assuming fixed-width 8-pixel characters */

  labelPos.y = chartBottom;
  for (m = Jan; m <= Dec; m++) {
    /* Find the center of this month's bar */
   x = MARGIN_WIDTH + m * mWidth + 0.5 * mWidth;

    /* Shift the label to the left by one-half its assumed length */
   labelPos.x = x - 0.5 * labelLength;
   pText (labelPos,  monthNames[m]);

    /* Get the coordinates for this month's bar */
   dataPos[0].x = dataPos[3].x = x - 0.5 * labelLength;
   dataPos[l].x = dataPos[2].x = x + 0.5 * labelLength;
   dataPos[0].y = dataPos[l].y = chartBottom + offset;
   dataPos[2].y = dataPos[3].y = chartBottom + offset + data[m];
   pFillArea (4, dataPos);
 }
}
```

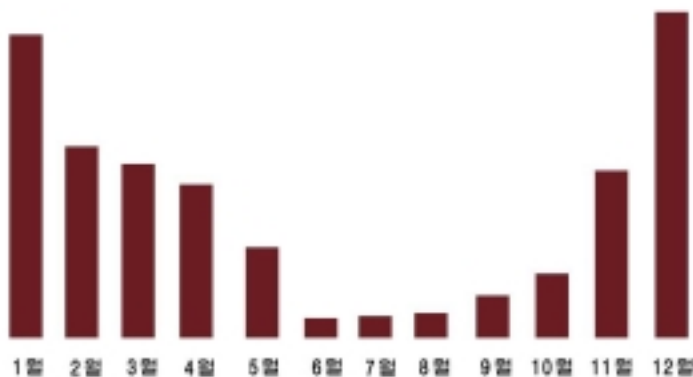



그림 3-53. barChart 프로그람으로 출력한 기둥도표의 현시

그림 3-54. pieChart 프로그람으로 만들어 낸 출력

쪼각원도표는 전체에 대한 개별적인 부분들의 퍼센트를 보여 주는데 사용된다.

다음의 프로그람은 수자와 입력에 의하여 결정되는 쪼각원들의 상대적인 크기를 가지고 쪼각원도표를 만든다. 이 프로그람으로 출력한 실례를 그림 3-54에 보여 주었다.

다음의 프로그람에서는 원의 방정식에서 약간한 변화를 준 결과를 출력하고 있다. 그림 3-55 에서 보여 준 형태들은 원의 반경 r를 변화시켜 만들었다.

r를 변화시키면서 그리면 라선형,심장형 달팽이형 기타 류사한 그림들을 만들어 낼수 있다.

그림 3-55. drawShape 프로그람으로 만들어 낸 곡선그림

```
#define TWO_PI 6.28

void pieChart (float * data)
{
 wcPt2 pts[2], center;
 float radius = WINDOW_HEIGHT / 4.0;
 float newSlice, total = 0.0, lastSlice = 0.0;
 Months month;

 center.x = WINDOW_WIDTH / 2;
 center.y = WINDOW_HEIGHT / 2;
 pCircle (center, radius);
 for (month = Jan; month <= Dec; month++)
   total += data[month];
 pts[0].x = center.x; pts[0].y = center.y;
 for (month = Jan; month <= Dec; month++) {
   newSlice = TWO_PI * data[month] / total + lastSlice;
   pts[1].x = center.x + radius * cosf (newSlice);
   pts[1].y = center.y + radius * sinf (newSlice);
   pPolyline (2, pts);
   lastSlice = newSlice;
 }
}
```

```
#include <stdio. h>
#include <math. h>
#include "graphics. h"

#define TWO_PI 6.28

/* Limacon equation is r = a * cos(theta) + b. Cardioid is the same,
```

```
    with a == b, so r = a * (1 + cos(theta)).
*/
typedef enum { spiral, cardioid, threeLeaf, fourLeaf, limacon } Fig;

void drawCurlyFig (Fig figure, wcPt2 pos, int * p)
{
 float r, theta = 0.0, dtheta = 1.0 / (float) p[0];
 int nPoints = (int) ceilf (TWO_PI * p[0]) + 1;
 wcPt2 * pt;

 if ((pt = (wcPt2 *) malloc (nPoints * sizeof (wcPt2))) == NULL) {
   fprintf (stderr, "Couldn't allocate points\ n");
   return;
 }

 /* Set first point for figure */
 pt [0].y = pos.y;
 switch (figure) {
 case spiral: pt[0].x = pos.x;                  break;
 case limacon: pt[0].x = pos.x + p[0] + p[1]; break;
 case cardioid: pt[0].x = pos.x + p[0] * 2;    break;
 case threeLeaf: pt[0].x = pos.x + p[0];       break;
 case fourLeaf: pt[0].x = pos.x + p[0];        break;
 }
 nPoints = 1;
 while (theta < TWO_PI) {
   switch (figure) {
   case spiral: r = p[0] * theta;    break;
   case limacon: r = p[0] * cosf (theta) + p[1]; break;
   case cardioid: r = p[0] * (1 + cosf (theta)); break;
   case threeLeaf: r = p[0] * cosf (3 * theta); break;
   case fourLeaf: r = p[0] * cosf (2 * theta); break;
   }
    pt[nPoints].x = pos.x + r * cosf (theta) ;
    pt[nPoints].y = pos.y + r * sinf (theta) ;
    nPoints++ ;
    theta += dtheta;
 }

 pPolyline (nPoints, pt);
 free (pt) ;
}

void main (int argc, char ** argv)
{
 long windowID = openGraphics (*argv, 400, 100);
 Fig f;
```

```
  /* Center positions for each figure */
  wcPt2 center[] = { 50, 50, 100, 50, 175, 50, 250, 50, 300, 50 };

  /* Parameters to define each figure. First four need one parameter.
     Fifth figure (limacon) needs two. */
  int p[5][2] = { 5, -1, 20, -1, 30, -1, 30, -1, 40, 10 } ;

  setBackground (WHITE) ;
  setColor (BLACK) ;
  for (f=spiral; f<=limacon; f++)
    drawCurlyFig (f, center[f], p[f]);
  sleep (10);
  closeGraphics (windowID) ;
}
```

참고문헌

브리센함알고리듬에 대한 정보는 Bresenham(1965,1977)에서 찾아 볼수 있다. 중점법에 대하여서는 Kappel(1985)을 보시오. 직선과 원을 발생시키기 위한 병렬적인 방법은 Pang(1990)과 Wright(1990)이 설명하고 있다. 보충적인 프로그람작성실례들과 PHIGS기초요소들에 대한 정보는 Howard(1991)들, Hopgood와 Duce(1991), Gaskins(1992), Blake(1993)에서 찾아 볼수 있다. GKS출력기초요소함수들에 대한 정보는 Hopgood(1983)들과 Enderle,Kansy, pfaff(1984)을 보시오.

련습문제

3-1. DDA알고리듬을 리용하여 polyline함수를 실현하시오. 입력점의 개수(n)는 임의로 주어진다. n=1일 때에는 하나의 점을 현시해야 한다.

3-2. 임의의 경사도를 가지는 직선을 발생시키도록 브리센함직선알고리듬을 확장하시오. 4분구들사이의 대칭관계를 고려하시오. 이 알고리듬을 써서 polyline함수를 n개 입력점들을 련결시키는 직선모임을 현시하는 하나의 루틴으로 실현하시오. n=1에 대하여서는 루틴이 하나의 점을 현시하게 한다.

3-3. 기하학적치수를 유지하는 수정된 브리센함선알고리듬(3장 10절)을 써서 선끝점들의 임의의 입력모임에 대해서도 일관성이 있는 polyline함수를 실현하기 위한 방안을 제기하시오.

3-4. 중점법을 써서 0< m <1 범위의 경사를 가지는 직선경로우의 점들을 발생시키기 위한 결심파라메터를 유도하시오. 그리고 중점결심파라메터가 브리센함직선알고리듬에서의 결심파라메터와 같은것이라는것을 보여 주시오.

3-5. 중점법을 써서 임의의 경사를 가지는 선분을 발생시키는데 쓸수 있는 결심파라메터를

유도하시오.

3-6. $0 < m < 1$ 범위의 경사도에 대한 브리센함선알고리듬의 병렬처리판을 만드시오.

3-7. 임의의 경사의 직선에 대한 브리센함알고리듬의 병렬처리판을 만드시오.

3-8. 인치당 100개 화소를 현시할수 있는 8×10in 영상현시장치가 달린 체계가 있다. 만약 기억기가 1byte 단어로 조직되여 있고 프레임완충기의 시작주소가 0이고 매 화소에 1byte 기억기가 할당되여 있다면 화면자리표 (x, y)인 화소의 프레임완충기의 주소는 얼마인가?

3-9. 인치당 100개 화소를 현시할수 있는 8×10in 영상현시장치가 달린 체계가 있다. 만약 기억기가 1byte 단어로 조직되여 있고 프레임완충기의 시작주소가 0이고 매 화소당 6bit 기ㄱ억기가 할당되여 있다면 화면자리표가 (x, y)인 화소의 프레임완충기의 주소(혹은 주소들)는 얼마인가?

3-10. 브리센함선알고리듬에서 setpixel루틴을 프레임완충기주소계산에 반복수법(3장 3절)을 써서 실현하시오

3-11. 원의 중점알고리듬을 기하학적치수가 유지되게(3장 10절) 현시되도록 수정하시오.

3-12. 원의 중점알고리듬의 병렬적실현을 위한 프로그람을 작성하시오.

3-13. 시작위치가 $(r_x, 0)$이고 점들이 시계바늘반대방향으로 곡선길을 따라 발생되게 된다고 보고 타원의 중점알고리듬에 대한 결심파라메터를 유도하시오.

3-14. 타원의 중점알고리듬의 병렬적실현을 위한 프로그람을 작성하시오.

3-15. 시누스함수를 현시하는데서 대칭성의 우점을 살리는 효과적인 알고리듬을 제기하시오.

3-16. 다음의 감쇠조화진동을 현시하기 위한 효과적인 알고리듬을 함수의 대칭을 고려하여 제기하시오.

$$y = Ae^{-kx}\sin(\omega x + \theta)$$

여기서 ω는 각주파수, θ는 시누스함수의 위상각이다.

시누스함수의 몇개 주기를 진폭값이 최대진폭값 A의 1/10로 떨어 질 때까지 x의 함수로서 그리시오.

3-17. 중점법을 써서 다음의 곡선을 $-10 \le x \le 10$구간에서 주사변환하기 위한 효과적인 알고리듬을 전개하시오. 대칭성을 고려하시오.

$$y = \frac{1}{12}x^3$$

3-18. 중점방법을 리용하고 대칭을 고려하여 다음의 포물선을 $-10 \le x \le 10$구간에서 주사변환하시오.

$$y = 100 - x^2$$

3-19. 포물선 $x=y^2$을 구간 $-10 \le x \le 10$에서 주사변환하는데 중점방법과 대칭고려방법을 쓰시오.

3-20. $y=ax^2 - b$ 형태의 임의의 포물선을 주사변환하기 위한 중점알고리듬을 작성하시오. 파라메터 a, b 그리고 x의 구간값은 입력된다.

3-21. 지적된 타원의 내부를 옹근색으로 주사변환하는 프로그람을 작성하시오.

3-22. 임의의 입력정점모임에 대하여 내부구역을 결정하기 위한 알고리듬을 령아닌 감음회수규칙을 써서 만드시오. 변이 지나가는 방향을 식별하기 위한 계산에는 벡토르적을 쓰시오.

3-23. 임의의 입력정점모임에 대하여 내부구역을 결정하기 위한 알고리듬을 령아닌 감음회수규칙을 써서 만드시오. 변이 지나가는 방향을 식별하기 위한 계산에는 스칼라적을 쓰시오.

3-24. 임의로 지적된 《다각형》정점모임의 내부를 채우기 위한 프로그람을 쓰시오. 내부구역식별에는 령아닌 감음회수규칙을 쓰시오.

3-25. 4련결구역에 대한 경계채우기알고리듬을 주사선법과 병합시켜 지나친 탄창조작을 피하도록 수정하시오.

3-26. 8련결구역을 채우기 위한 경계채우기프로그람을 쓰시오.

3-27. 중점법을 써서 현시된 타원을 경계채우기알고리듬으로 어떻게 정확히 채울수 있는가를 설명하시오.

3-28. 임의의 지적된 구역의 내부를 채우기 위한 침수채우기알고리듬을 전개하고 실현하시오.

3-29. text함수를 실현하기 위한 루틴을 쓰시오.

3-30. polymarker함수를 실현하기 위한 루틴을 쓰시오.

3-31. polyline함수를 써서 기둥도표를 현시하기 위한 프로그람을 쓰시오. 프로그람에 대한 입력은 자료점들과 x축과 y축에 요구되는 표식이다. 자료점들은 그라프가 전체 화면구역에 걸쳐 현시되도록 프로그람에 의하여 비례변환되여야 한다.

3-32. 기둥도표를 임의의 지정된 화면구역에 현시하기 위한 프로그람을 쓰시오. 기둥을 그리는데 polyline함수를 사용하시오.

3-33. 임의의 입력자료점모임에 대한 선그라프를 임의로 선택된 화면구역에 현시하기 위한 프로그람을 쓰시오. 입력자료모임은 선택된 화면구역을 채우도록 비례변환한다. 자료입력점들은 선분과 결합된 별표(*)로 현시되여야 하며 x 축과 y 축은 입력내용에 따라 표식되여야 한다(자료점을 그리는데 별표대신에 작은 동그라미나 다른 몇가지 기호들을 리용할수도 있다.).

3-34. circle함수를 리용하여 적당한 표식이 있는 쪼각원도표를 현시하기 위한 루틴을 쓰시오. 루틴에 대한 입력은 몇개의 구간에서의 자료의 분포를 주는 자료모임, 쪼각원도표의 이름, 구간들의 이름이다. 매 구역표식은 쪼각원도표경계밖의 해당한 쪼각원부분가까이에 현시되여야 한다.

4장. 출력기초요소들의 속 성

일반적으로 기초요소들의 현시방식에 영향을 미치는 파라메터를 속성파라메터라고 한다. 색과 크기가 같은 일부 속성파라메터들은 기초요소의 기본적인 특성을 결정한다. 다른것들은 특별한 조건하에서 기초요소들이 어떻게 현시되는가를 지적한다. 이 부류의 속성실례에는 3차원보기에서의 깊이정보와 대화식물체선택프로그람에서의 보임성 또는 검출가능성이 속한다. 이 특수조건속성들은 뒤장에서 고찰한다. 여기서는 특별한 조건에 관계없이 기초요소들의 기본적인 현시특성을 조종하는 속성들만 고찰한다. 실례로 직선은 점선 또는 파선, 굵은선 또는 가는선, 푸른색선 또는 감색선이 될수 있을것이다. 구역은 단색 또는 다색무늬로 채워 질수 있다. 본문은 왼쪽에서 오른쪽으로, 화면의 대각선을 따라서 경사 지게 또는 수직렬로 씌여 질수 있다. 개별적인 문자들은 서로 다른 폰트, 색, 크기로 현시될수 있다. 그리고 또 라스터계단효과를 원활하게 하기 위하여 물체의 경계에서 세기변화를 줄수도 있다.

속성선택항목들을 도형처리프로그람에 병합하기 위한 한가지 방법은 매개 출력기초요소들의 함수에 관계되는 파라메터목록을 해당한 속성이 포함되도록 확장하는것이다. 실례로 직선긋기함수에는 끝점들의 자리표외에 색, 너비 기타 속성들을 설정하는 파라메터들을 포함시킬수 있다. 다른 방법은 체계의 현재 속성값들의 목록을 가지고 있는것이다. 그리고 속성목록안에서 현재값을 설정하기 위한 개별적인 함수들을 도형처리프로그람에 포함시킨다. 출력기초요소들을 발생시키기 위하여 체계는 관계되는 속성들을 검사하고 현재의 속성설정을 리용하여 기초요소들에 대한 현시루틴을 호출한다. 일부 프로그람들은 출력기초요소명령에서 속성함수와 속성파라메터의 결합을 사용자에게 제공하고 있다. GKS와 PHIGS표준에서는 속성설정이 체계의 속성목록을 갱신하는 개별함수들에 의하여 진행된다.

1절. 직선의 속성

선분의 기본적인 속성은 형태, 너비, 색이다. 일부 도형처리프로그람들에서는 선을 펜 또는 붓선택항목을 리용하여 현시할수도 있다. 아래에서는 선긋기루틴이 여러가지 속성지적에 편리하도록 어떻게 수정될수 있는가를 고찰한다.

직선의 형태

직선의 형태속성에서 선택가능한것은 실선, 파선, 점선들이다. 이러한 선들을 발생시키기 위하여 선경로를 따라 현시되는 긋기부분의 길이와 간격을 설정하게끔 직선긋기알고리듬을 수정한다. 파선은 긋기부분과 간격을 길이가 같게 발생시켜 현시할수 있다. 긋기부분길이와 그사이 간격의 길이는 흔히 사용자의 선택항목으로 지적된다. 점선은 긋기부분의 길이는 대단히 짧고 상대적으로 간격은 크게 하여 현시할수 있다. 이러한 방법은 다른 형태의 직선을 만드는데도 리용된다.

PHIGS응용프로그람에서 직선의 형태속성을 설정하기 위하여 사용자는 함수

```
setLinetype (lt)
```

를 호출한다. 여기서 파라메터 lt에는 각각 실선, 파선, 점선, 한점쇄선을 발생시키는 정의 옹근수값 1, 2, 3, 4가 할당된다. 직선의 형태파라메터에 대한 다른 값들은 여러가지 선형태를 현시하는데 리용될수 있다. PHIGS응용프로그람에서 직선의 형파라메터가 설정되면 그후의 모든 직선긋기명령은 이 형태의 직선을 만든다. 다음의 프로그람토막은 그림 4-1의 자료그라프를 현시하기 위하여 리용한 Linetype명령을 설명하고 있다.

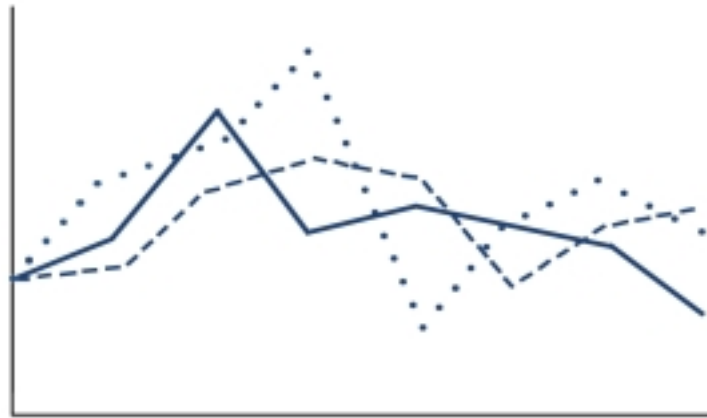



그림 4-1. 프로그람 chartData 에 의한 출력(세개 자료모임을 서로 다른 세가지 직선형태을 가지고 현시)

```
#include <stdio.h>
#include "graphics.h"

#define MARGIN_WIDTH 0.05 * WINDOW_WIDTH

int readData (char * inFile, float * data)
{
 int fileError = FALSE;
 FILE * fp;
 int month;

 if ((fp = fopen (inFile, "r")) == NULL)
   fileError = TRUE;
 else {
   for (month=0; month<12; month++)
     fscanf (fp, "%f", &data[month]);
   fclose (fp) ;
 }
 return (fileError) ;
}

void chartData (float * data, pLineType lineType)
{
 wcPt2 pts[12] ;
 float monthWidth = (WINDOW_WIDTH - 2 * MARGIN_WIDTH) / 12;
 int i ;

 for (i=0; i<12; i++) {
   pts[i].x = MARGIN_WIDTH + i * monthWidth + 0.5 * monthWidth;
   pts[i].y = data[i] ;
```

```
    }
    pSetLineType (lineType);
    pPolyline (12, pts);
   }

   int main (int argc, char ** argv)
   {
    long windowID = openGraphics (*argv, WINDOW_WIDTH, WINDOW_HEIGHT);
    float data[12] ;

    setBackground (WHITE) ;
    setColor (BLUE) ;
    readData ("../data/data1960", data);
    chartData (data, SOLID) ;
    readData ("../data/data1970", data);
    chartData (data, DASHED) ;
    readData ("../data/data1980", data);
    chartData (data, DOTTED) ;
    sleep (10) ;
    closeGraphics (windowID) ;
   }
```

라스터선알고리듬은 화소구간을 그리는 방법으로 선의 형태속성을 현시한다. 여러가지 파선, 점선, 한점쇄선들에 대하여 선그리기절차는 긋기부분들사이의 여러 화소들을 뛰여 넘으면서 선경로우의 련속된 화소토막들을 출력한다. 긋기부분의 길이와 간격에 해당한 화소의 개수는 화소마스크로 지적할수 있다. 화소마스크는 선경로우에서 어느 위치가 현시되는가를 지적하는 수자 1과 0으로 된 렬이다.

실례로 마스크 1111000은 긋기부분의 길이가 4화소, 사이간격이 3화소인 파선을 현시하는데 리용될수 있다. 2값준위체계에서는 마스크가 선택된 선형태를 현시하기 위하여 선경로를 따라 프레임완충기에 적재되여야 할 비트값을 준다.

화소의 개수를 고정시켜 긋기부분을 현시하면 그림 4-2에서 보여 준바와 같이 서로 다른 방향으로 놓여 있는 선들에서 긋기부분들이 길이가 다르게 된다. 이 그림에서 긋기부분들은 다같이 4개의 화소로 현시되여 있지만 대각방향에서는 $\sqrt{2}$배 더 길다. 정확히 그리자면 긋기부분의 길이가 어느 방향에서나 일정하게 되여야 한다. 그러자면 긋기부분과 사이간격에 대한 화소의 개수를 선경사도에 따라 조정해야 한다. 그림 4-2에서 만일 대각선방향의 긋기부분을 3개 화소로 줄이면 거의 같은 길이의 긋기부분들을 현시할수 있다. 긋기부분의 길이를 유지하기 위한 다른 방법은 긋기부분을 개별적인 선분으로 취급하는것이다. 매 긋기부분에 대한 끝점자리표들의 위치를 지적하여 선루틴에 넘겨 주면 긋기부분의 경로를 따라 화소위치들을 계산한다.

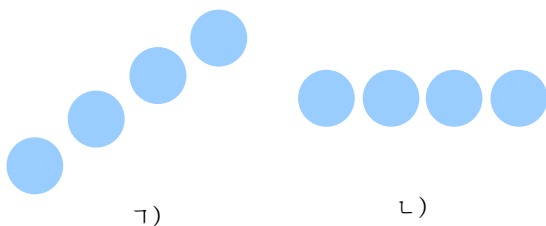



그림 4-2. 화소의 개수는 같지만 길이가 다르게 현시되는 긋기부분

직선의 너비

선너비선택항목의 실현은 출력장치의 능력에 관계된다. 영상현시장치우에서 두꺼운 선은 린접한 평행선들로 현시되게 되며 한편 펜작도기에서는 펜을 바꿀것을 요구할수 있다. 다른 PHIGS속성들과 마찬가지로 선너비명령은 속성목록의 현재의 선너비값을 설정하는데 리용된다. 이 값은 후에 선긋기 알고리듬에서 출력기초요소명령으로 만들어 지는 선의 두께를 조정하는데 리용된다.

선너비속성은

```
setLinewidthScaleFactor (lw)
```

명령에 의하여 설정한다. 선너비파라메터 lw에는 현시될 선의 너비를 지적하는 정의 수가 할당된다. 값 1은 표준너비선을 가리킨다. 실례로 펜작도기에서 사용자가 표준선너비의 절반인 선을 그리려면 lw를 값 0.5로 설정하면 된다. 1보다 큰 값은 표준보다 두꺼운 선을 만든다.

라스터적실현에서 표준너비선은 브리센함의 알고리듬에서와 같이 매 표본위치에서 하나의 화소에 의해 만들어 진다. 다른 너비선은 린접한 평행선경로를 따라 추가적인 화소들을 현시함으로써 표준선의 정의 옹근수배로 현시된다. 경사도절대값이 1보다 작은 경우 두꺼운 선을 현시하기 위하여서는 선의 매 위치에서 화소들의 수직구간을 현시하도록 선긋기루틴을 수정할수 있다. 매개 구간의 화소의 개수는 파라메터 lw의 옹근수크기와 같도록 설정한다. 그림 4-3에서는 본래의 선경로에 평행인 선을 발생시켜 2중너비선을 긋는다. 매개 x표본위치에서 대응하는 y자리표를 계산하고 화면자리표가 (x, y) 및 $(x, y+1)$ 인 화소들을 현시한다. $lw \geq 3$인 선은 단일너비선경로의 우와 아래의 화소들을 현시한다.

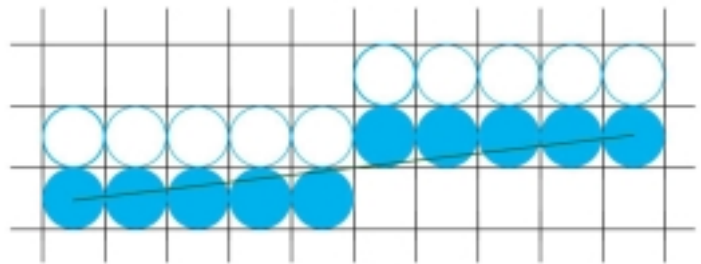



그림 4-3. 수직화소구간에 의하여 그려 지는 경사도가 $|m|<1$ 인 2중너비라스터선

경사도절대값이 1보다 큰 경우에는 선을 선경로의 오른쪽과 왼쪽에서 화소들을 선택하는 수평구간을 가지고 두꺼운 선을 그을수 있다. 이 방법을 그림 4-4에서 보여 주고 있는데 여기에는 너비가 4인 선이 수평화소구간에 의하여 그려 져 있다.

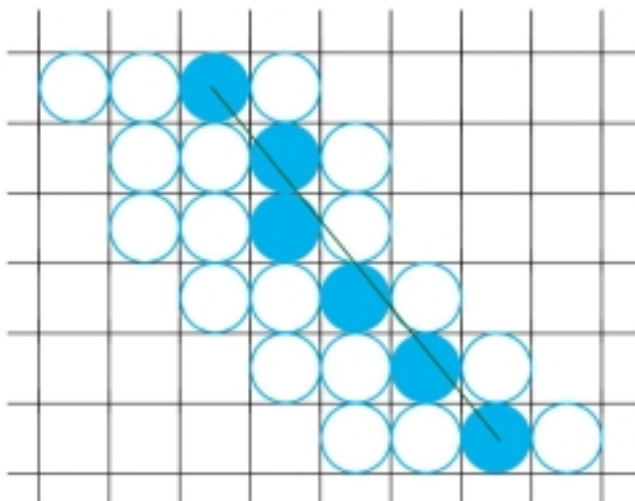



그림 4-4. 수평화소구간에 의하여 그려 지는 경사도가 $|m|>1$ 이고 선너비파라메터 lw=4 인 라스터선

비록 수평 및 수직화소구간을 가지고 두꺼운 선을 현시하면 빠르지만 현시되는 선의 너비(선경로에 수직으로 측정되는)가 선의 경사도에 따라 달라 지게 된다. 45°선은 같은 길이의 화소구간에 의하여 현시되는 수평 또는 수직선에 비하여 결수 $1/\sqrt{2}$로 가늘게 현시되게 된다.

수평 혹은 수직화소구간을 리용하여 너비선택항목을 실현하는데서 다른 하나의 문제는 이 방법이 선의 경사도에 무관계하게 끝들이 수평 또는 수직으로 잘라 지는것이다. 이 결과는 대단히 두꺼운 선에서 더 눈에 뜨이게 된다. 선끝의 형태는 거기에 선모자 (line cap)

를 추가하는 방법으로 보기 좋게 현시되도록 조정할수 있다(그림4-5). 선모자의 한가지 형태는 두꺼운 선을 선경로에 수직인 바른4각형끝을 가지게 현시되도록 요소평행선들의 끝위치를 조정하는 절단모자(butt cap)이다. 지적된 선이 경사도 m을 가지면 두꺼운 선의 바른4각형끝은 경사도 $-1/m$을 가진다. 다른 선모자는 매 절단모자에 채워 진 반원을 추가하는 둥근모자(round cap)이다. 이 원호는 중심이 선끝점에 있으며 선의 두께와 같은 직경을 가진다. 세번째 형의 선모자는 바른4각형투영모자(projecting square cap)이다. 여기서는 선을 간단히 지적된 끝점들을 지나서 선너비의 절반위치까지 연장한 위치에서 절단모자를 추가한다.

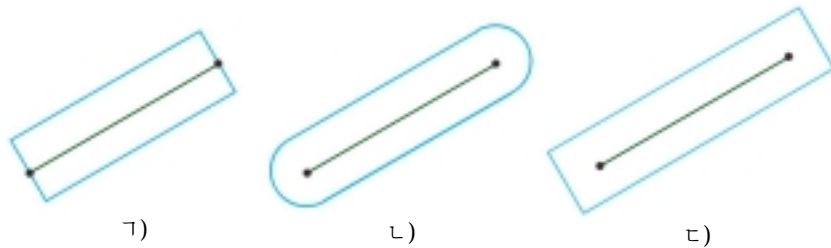



그림 4-5. 절단모자(ㄱ), 둥근모자(ㄴ), 바른4각형투영모자(ㄷ)로 그려 진 두꺼운 선

두꺼운 선을 만드는 기타 방법들에는 선을 채워 진 직4각형처럼 현시하는 방법 또는 다음부분에서 설명하는것처럼 선택된 펜 또는 붓무늬로 선을 발생시키는 방법이 있다. 선경계에 대한 직4각형표현을 얻기 위하여서는 정점자리표들이 선경로에 수직이면서 선끝점으로부터 선너비절반만큼 옮겨지도록 직4각형정점들의 위치를 계산한다. 그러면 직4각형선은 그림 4-5 ㄱ와 같이 나타난다. 그다음에 채워 진 직4각형에 둥근모자를 추가하거나 또는 바른4각형투영모자를 현시하기 위하여 길이를 연장할수 있다.

두꺼운 절선의 발생은 몇가지 추가적인 고찰을 요구한다. 일반적으로 하나의 선토막현시에서 고찰한 방법들로는 원활하게 련결되는 선토막들의 련속을 얻지 못한다. 실례로 수평 및 수직화소구간을 리용하여 두꺼운 절선을 현시하면 수평구간으로부터 수직구간에로 이행하는 곳에서 서로 다른 경사도의 선들사이의 경계에 화소터짐이 생긴다. 토막들의 끝점에서 추가적인 처리를 진행하여야 원활하게 련결된 두꺼운 절선을 만들수 있다. 그림 4-6은 두 선토막을 원활하게 련결하는 세가지 가능한 방법을 보여 준다. 연귀련결(miter join)은 두선의 매 바깥경계를 그것들이 만날 때까지 연장하는 방법으로 진행한다. 둥근련결(round join)은 두 토막사이를 직경이 선너비와 같은 원경계로 모자를 씌우는 방법으로 진행한다. 경사련결(bevel join)은 절단모자를 가지는 선토막을 현시하고 토막들이 만나는 곳에서 3각형홈을 채우는 식으로 진행한다. 만약 련결된 두 선토막사이의 각이 대단히 작으면 연귀련결은 절선의 모양을 외곡하는 긴 못을 발생시킬수 있다. 어떤 도형처리프로그람에서는 두개의 련결된 토막이 충분히 작은 각으로 만나는 경우 연귀련결로부터 경사련결로 바꾸도록 하여 이 현상을 피하게 한다.

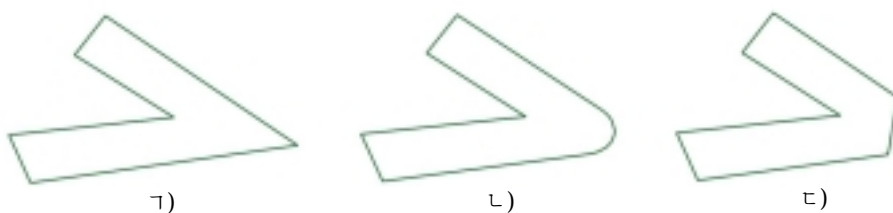



그림 4-6. 연귀련결(ㄱ), 둥근련결(ㄴ), 경사련결(ㄷ)로 이어 진 두꺼운 선

펜과 붓의 선택항목

일부 프로그람들에서는 펜 또는 붓을 선택하여 선을 현시한다. 이런 종류의 선택항목에는 형태, 크기, 무늬가 들어 간다. 가능한 몇가지 펜 또는 붓의 형태를 그림 4-7에 보여 주었다. 이 형태들은 선경로를 따라 설치되여야 할 화소위치들의 배렬과 관련되는 화소마스크안에 기억되게 된다. 실례로 직4각형펜은 그림 4-8에 보여 준 마스크로 그림 4-9에서와 같이 선경로를 따라 마스크의 중심(또는 한쪽 구석)을 움직이면 실현할수 있다. 프레임완충기에서 화소가 한번이상 설정되는것을 피하기 위하여 간단히 마스크의 매 위치에서 발생되는 수평구간들을 모아 놓고 매 주사선을 따르는 구간들에 대한 시작 및 끝의 x위치를 추적할 수 있다.

그림 4-7. 선현시를 위한 펜 및 붓형

마스크의 크기를 변화시키면 펜(또는 붓)형태로 만들어 지는 선을 여러가지 너비로 현시할수 있다. 실례로 그림 4-9의 직4각형펜선은 2×2 직4각형마스크로 좁게 하거나 4×4마스크로 넓힐수 있다. 또한 선은 무늬값을 펜 또는 붓마스크에 덧놓아 해당한 무늬로 현시할수 있다. 선무늬의 일부 실례를 그림 4-10에 보여 주었다. 그리기프로그람에서 제공될수 있는 추가적인 무늬선택항목은 모의붓놀림을 현시하는것이다. 그림 4-11은 각이한 형의 붓놀림을 모의하여 현시한 일부 무늬를 보여 주고 있다.

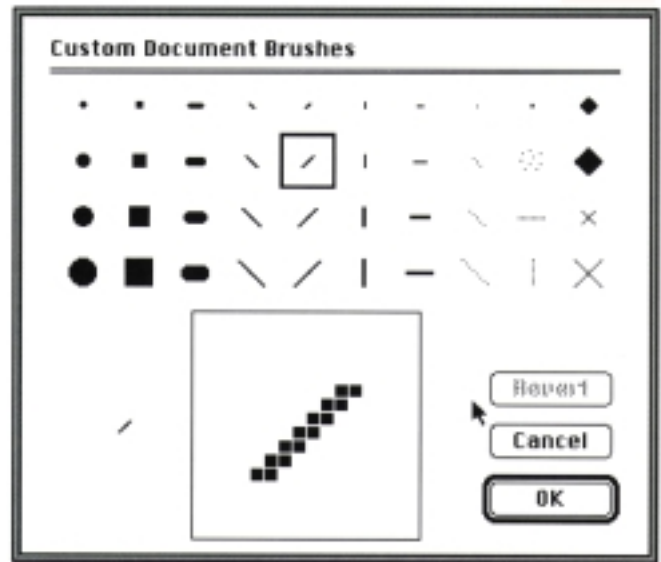
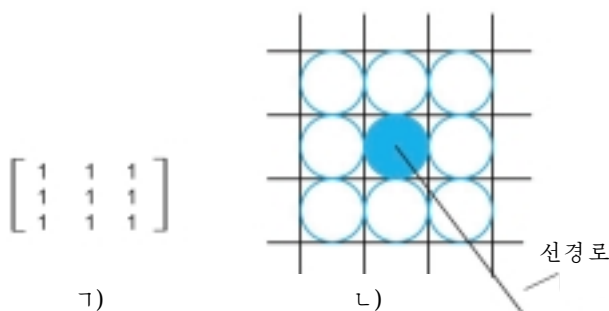



그림 4-8. 직4각형펜에 대한 화소마스크, 마스크를 지적된 화소위치에 중심을 맞추어 놓을 때 련관된 화소배렬

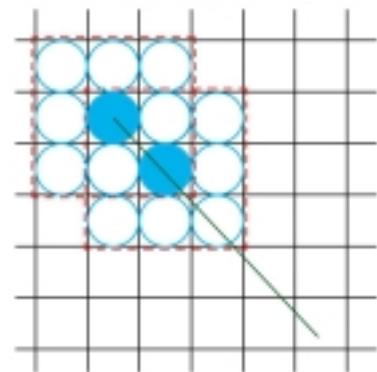



그림 4-9. 그림 4-8의 펜형태에 의한 선발생

그림 4-10. 여러가지 형태와 무늬를 리용하는 그리기프로그람으로 그은 곡선(왼쪽에서 오른쪽으로 붓형태는 바른4각형, 원형, 대각선형, 점무늬형, 분무형이다.)

직선의 색

체계가 색(또는 세기)선택항목을 제공하는 경우 현재의 색첨수를 주는 파라메터가 체계의 속성값 목록에 포함된다. 절선루틴은 setPixel함수를 리용하여 프레임완충기의 선경로우의 화소위치들에 이 색값을 설정하여 선을 현재색으로 현시한다.

그림 4-11. 다루마인형(일본에서 행운의 표시); 콤퓨터미술가 고이시 고자끼가 페인트브라쉬체계를 리용하여 그렸다. 원래 다루마인형은 처음에 눈이 그려 져 있지 않다. 한쪽 눈은 어떤 소원을 가지고 있을 때 그려 넣으며 다른 눈은 그 소원이 성취되였을 때 그린다.

선택할수 있는 색의 개수는 프레임완충기의 매 화소에서 사용할수 있는 비트수에 관계된다.

PHIGS에서는 선의 색값을 함수

```
setPolylineColourIndex(lc)
```

에 의하여 설정한다. 선택하여 쓸수 있는 색에 대응하는 부아닌 옹근수값이 선택파라메터 lc에 할당된다. 배경색으로 그려 진 선은 보이지 않는다. 사용자는 이전에 현시된 선을 배경색으로 재설정하면 지울수 있다(선이 배경색구역과 한번이상 겹치지 않는다고 가정하고).

응용프로그람에서 여러가지 선속성명령을 리용하는 실례를 다음의 명령문들에 의하여 준다.

```
setLinetype (2);
setLinewidthScaleFactor (2);
setPolylineColourIndex (5);
Polyline (n1, wcpoints1);

setPolylineColourIndex (6);
Polyline (n1, wcpoints2);
```

이 프로그람토막은 2중너비, 파선으로 그어 지는 두개의 도형을 현시한다. 첫번째것은 색값 5에 대응하는 색으로 현시되며 두번째것은 색 6으로 현시된다.

2절. 곡선의 속성

곡선의 속성파라메터들은 선분에 대한것과 같다. 곡선은 여러가지 색, 너비, 선무늬, 펜 또는 붓선택항목들로 현시할수 있다. 곡선그리기알고리듬을 속성선택을 할수 있게 적응시키는 방법은 직선긋기에 대한것과 류사하다.

선의 형선택항목을 실현할 때 론의된 화소마스크는 파선 및 점선무늬를 발생시키기 위한 라스터곡선알고리듬에도 리용된다. 실례로 마스크 11100은 그림 4-12에 보여 준 파선으로 된 원을 만든다. 다른 8분구에서의 파선은 원대칭을 리용하여 발생시킬수도 있지만 한 8분구에서 다른데로 움직일 때 긋기부분과 간격의 순서를 정확히 유지하려면 화소위치들을 밀기하여야 한다. 또한 직선알고리듬에서와 같이 화소마스크는 곡선의 경사도에 따라 긋기부분과 간격의 길이가 달라 지게 한다. 긋기부분과 간격의 길이를 일정하게 하려면 원둘레를 움직일 때 매 긋기부분에서 현시되는 화소의 수를 조정할 필요가 있다. 길이가 같은 파선을 만들기 위하여 고정된 화소구간을 가지는 화소마스크대신에 동일한 각도의 호를 따라 화소를 현시한다.

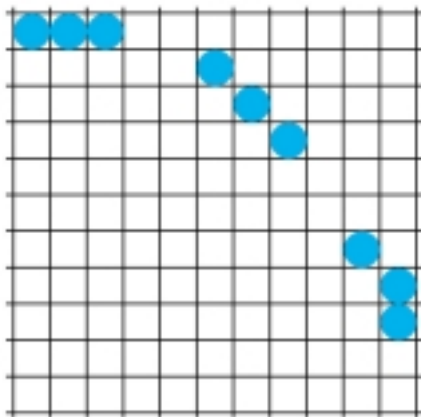



그림 4-12. 긋기부분 3 화소, 사이 간격 2 화소로 현시된 원호

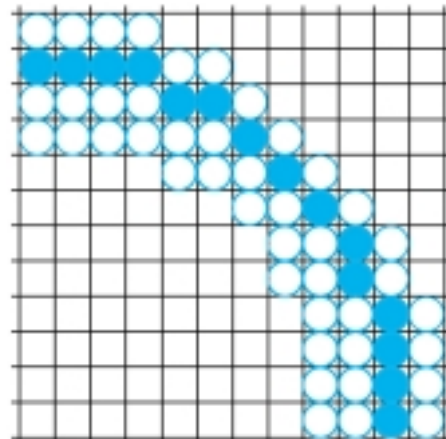



그림 4-13. 화소구간으로 그린 너비가 4 인 원호

여러가지 너비의 라스터곡선들은 수평 또는 수직화소구간방법을 리용하여 현시할수 있다. 곡선경사도의 절대값이 1보다 작은 곳에서는 수직구간을 현시하고 경사도의 절대값이 1보다 큰 곳에서는 수평구간을 현시한다. 그림 4-13에서는 첫 4분구에서 너비가 4인 원호현시에 대한 이 방법을 보여주었다. 원대칭을 리용하여 x=0부터 x=y까지의 8분구에서 수직구간을 가지고 원경로를 발생시킨 다음곡선의 나머지를 얻기 위하여 화소위치들을 선 y= x에 대하여 반사시킨다. 다른 4분구들에서의 부분원은 첫 4분구에서의 화소위치들을 자리표축에 대하여 반사시키면 얻어 진다. 이 방법에 의하여 현시되는 곡선의 두께도 역시 곡선경사도의 함수이다. 원, 타원 기타 곡선들은 경사도가 절대값 1을 가지는 곳에서 제일 가늘게 나타나게 된다.

두꺼운 곡선을 현시하는 다른 방법은 두개의 평행인 곡선경로사이의 구역을 채우는것이다. 그것들사이의 거리는 요구되는 너비와 같다. 이것은 지적된 곡선경로를 하나의 경계로 리용하고 두번째 경계는 본래곡선경로의 내부 또는 외부에 설정할수 있다. 그러나 이 방법은 두번째 경계를 선택하는 방향에 따라 본래의 곡선경로를 안쪽 또는 바깥쪽으로 밀게 된다. 두개의 경계곡선을 지적된 곡선경로의 량쪽으로 1/2너비거리에 설정하면 본래의 곡선위치를 유지할수 있다.

이 방법의 실례를 그림 4-14에서 반경이 16이고 너비가 4인 원토막에 대하여 보여 주었다. 이때 경계호들은 반경 16의 량쪽으로 거리 2만큼 떨어 진 곳에 설정된다. 원호의 정확한 크기를 유지하기 위하여서는 3장 10절에서 설명된바와 같이 동심경계호들에 대한 반경을 r=14와 r =17로 설정한다.

이 방법은 두꺼운 원에 대하여서는 정확하지만 일반적으로 기타 다른 두꺼운 곡선에 대하여서는 실제구역에 대한 근사를 준다. 실례로 이 방법에 의하여 발생시킨 둥그런 타원의 내부 및 외부경계는 동일한 초점을 가지지 않는다.

곡선의 펜(또는 붓)현시는 선분에서 설명한것과 같은 기술을 리용하여 발생시킨다. 그림 4-15의 첫 4분구에서의 원호에서 보여 준것처럼 선경로를 따라 펜형태를 복사한다. 여기서 직4각형펜의 중심은 곡선위치를 따라 련속 움직이면서 그림의 곡선모양을 만든다. 직4각형펜을 가지고 이 방법으로 현시되는 곡선은 곡선경사도의 절대값이 1인 곳에서 더 두꺼워 지게 된다. 곡선의 두께를 일정하게 하려면 직4각형펜을 곡선주위로 움직일 때 경사도방향으로 놓이도록 회전시키거나 또는 원형펜을 리용하면 현시할수 있다. 펜과 붓형태로 그려 지는 곡선은 여러가지 크기, 무늬, 붓놀림을 모의하여 현시할수 있다.

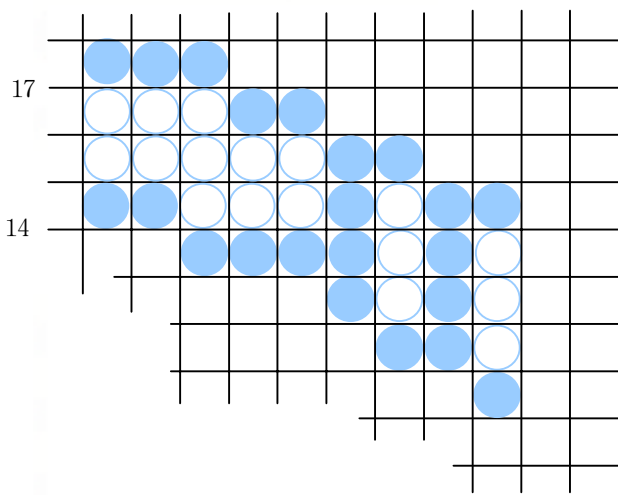



그림 4-14. 두개의 동심원호사이의 구역을 채워 현시한 너비가 4이고 반경이 16인 원호

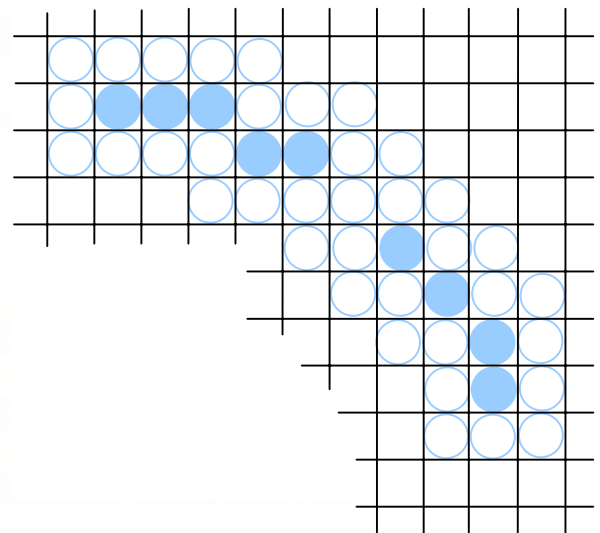



그림 4-15. 직4각형펜으로 현시되는 원호

3절. 색 및 회색계조준위

여러가지 색 및 세기준위선택항목은 개별적인 체계의 능력과 설계목적에 따라 사용자가 사용할수 있게 되여 있다. 실례로 일반목적라스터주사체계는 보통 넓은 범위의 색을 제공하고 있는데 우연주사현시장치는 대표적인 몇가지 색만을 선택할수 있게 되여 있다. 색선택항목은 0부터 정의 옹근수범위의 값들에 의하여 수자적으로 부호화되여 있다. 그다음 CRT현시장치에서 이 색코드들이 전자속에 대한 세기준위설정으로 변환된다. 색작도기에서는 코드가 잉크분사 또는 펜선택을 조종하게 된다.

색라스터체계에서 선택가능한 색의 수는 프레임완충기에서 매 화소에 배당되는 기억크기에 관계된다. 또한 색정보는 프레임완충기에 두가지 방법으로 기억될수 있다. 색코드를 프레임완충기에 직접 기억시킬수도 있고 색코드를 개별적인 표에 넣고 화소값을 이 표안에서 첨수로 리용할수도 있다. 직접 기억시키는 방법에서는 응용프로그람의 색코드가 개별적으로 지적될 때마다 대응되는 2진수값

을 프레임완충기안에서 그 색으로 현시될 화소위치에 넣는다. 이 방법에서 색의 최소수는 표 4-1에 보여 준바와 같이 화소당 3bit의 기억기로 제공될수 있다. 매 비트는 RGB현시장치에서 대응하는 전자총의 세기준위(on 또는 off)를 조종하는데 리용된다. 제일 왼쪽 비트는 붉은색총을 조종하고 중간비트는 풀색총을 조종하며 제일 오른쪽 비트는 푸른색총을 조종한다. 프레임완충기에서 화소당 더 많은 비트를 할당하면 선택할수 있는 색의 수가 증가된다. 화소당 6bit인 경우에는 매총에 대하여 2bit씩 리용할수 있다. 이것은 매개 색총에 대하여 4개의 서로 다른 세기설정을 할수 있게 하며 화면의 매개 화소에 대하여 총 64가지의 색값을 사용할수 있게 한다. 분해능이 1024×1024인 완전색(화소당 24bit)RGB체계는 프레임완충기에 대하여 3MB의 기억공간을 요구한다. 색표는 큰 프레임완충기를 요구함이 없이 사용자에게 많은 색을 제공하는 또 하나의 방법이다. 대체로 개인용콤퓨터체계들에서는 가격을 낮추기 위하여 색표를 리용하여 프레임완충기의 기억기를 줄이고 있다.

표 4-1. 화소당 3bit 프레임완충기에 대한 8가지 색코드

색	프레임완충기에 기억되는 색값			현시되는 색
코드	붉은색	풀색	푸른색	
0	0	0	0	검은색
1	0	0	1	푸른색
2	0	1	0	풀색
3	0	1	1	푸른풀색
4	1	0	0	붉은색
5	1	0	1	분홍색
6	1	1	0	누런색
7	1	1	1	흰색

색표

그림 4-16에서는 **색검색표**(또는 **영상검색표**)에 색값을 기억시키는 방법을 보여 주고 있다. 여기서 프레임완충기의 값은 색표의 첨수로 리용한다. 이 실례에서 매개 화소는 256개의 표위치중 임의의 하나를 참조할수 있으며 표안의 매개 항목값은 색을 지적하는데 24bit를 리용한다. 색코드 2081에 대하여서는 풀색-푸른색결합이 화소위치 (x, y)에 현시된다. 이런 검색표를 쓰면 체계는 사용자에게 약 1700만색의 조색판으로부터 동시에 현시할수 있는 임의의 256가지 색을 조합할수 있다. 이 방법은 완전색체계에 비하여 동시에 현시할수 있는 색의 개수가 적지만 프레임완충기에 대한 기억기요구를 1MB로 줄인다. 일부 도형처리체계들은 프레임완충기에 화소당 9bit를 제공함으로써 사용자가 매 현시에서 512가지 색을 선택리용할수 있게 한다.

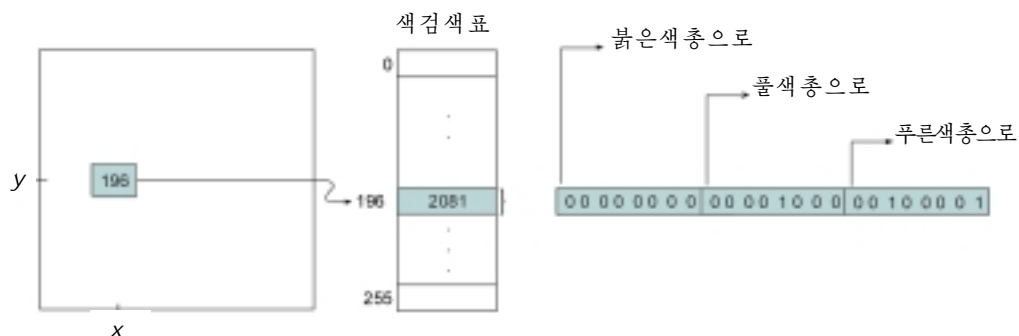



그림 4-16. 화소당 8bit의 프레임완충기로부터 호출되는 항목값당 24bit길이의 색검색표
(화소위치(x, y)에 기억된 값 196은 이 표에서 값 2081을 가지는 위치를 가리킨다.
이 항목값의 매 8bit토막이 RGB현시장치에서 각 전자총의 세기준위를 조종한다.)

사용자는 PHIGS응용프로그람에서 색표의 항목값을 함수

```
setColourRepresentation (ws, ci, colorptr)
```

에 의하여 설정할수 있다. 파라메터 ws는 워크스테이션출력장치를 가리킨다. 파라메터 ci는 색첨수를 가리키는데 그것은 색표위치번호(그림 4-16의 실례에서 0~255)이다. 그리고 파라메터 colorptr는 매개가 0~1사이 범위에서 지적되는 RGB색값(r, g, b)을 가리킨다. 색현시장치에 대하여 할수 있는 표항목값의 실례를 그림 4-17에 주었다.

ws=1

ci	색
0	(0, 0, 0)
1	(0, 0, 0.2)
.	.
.	.
.	.
192	(0, 0.03, 0.13)
.	.
.	.
.	.

ws=2

ci	색
0	(1, 1, 1)
1	(0.9, 1, 1)
2	(0.8, 1, 1)
.	
.	
.	

그림 4-17. 워크스테이션색표

색코드를 검색표에 기억시키면 여러가지 우점이 있다. 색표를 리용하면 큰 프레임완충기를 요구함이 없이 《적당한》 개수의 동시색을 제공할수 있다. 대부분의 응용들에서는 하나의 그림에서 256 또는 512가지의 서로 다른 색이면 충분하다. 또한 표항목값들은 임의의 시각에 변화시킬수 있다. 이것은 사용자가 도형처리자료구조에 대한 속성설정을 변화시킴이 없이 설계, 장면, 그라프에서 서로 다른 색결합을 쉽게 시험해 볼수 있게 한다. 류사하게 가시화응용들에서는 프레임완충기에 에네르기와 같은 일부 물리적량에 대한 값을 기억시킬수 있는데 화소값을 변화시키지 않고 여러가지 색부호화를 시험해 보는데 검색표를 리용할수 있다.

그리고 가시화 및 화상처리응용들에서 색표방법은 지적된 턱값의 우 또는 아래의 모든 화소값들이 동일한 색으로 설정될수 있도록 색턱값을 설정하는데서 편리한 방법이다. 이런 리유로 일부 체계들은 색코드를 기억시키는 두가지 방법을 다 제공하여 사용자가 색표를 리용하거나 프레임완충기에 색코드를 직접 기억시키는 방법을 선택할수 있게 하고 있다.

회색계조

색현시능력이 없는 현시장치들에서 색함수는 응용프로그람에서 현시되는 기초요소들에 대하여 회색의 명암 또는 회색계조를 설정하는데 리용될수 있다. 0~1사이의 수값이 회색계조준위를 지적하는데 리용된다. 이것은 다시 해당한 2진코드로 변환되여 라스터기억기에 기억된다. 이렇게 함으로써 세기설정이 다른 회색계조능력을 가지는 체계들에 쉽게 적응시킬수 있게 한다.

표 4-2는 4준위회색계조체계의 세기코드에 대한 값을 보여 준다. 이 실례에서 0.33에 가까운 모든 세기항목값은 프레임완충기에 2진수 01로 기억되며 이 값을 가지는 화소는 진한 회색으로 현시되게 된다. 화소당 추가적인 비트들을 프레임완충기에서 쓸수 있으면 값 0.33은 제일 가까운 준위에로 넘어 가게 된다. 화소당 3bit이면 8개의 회색준위를 조정할수 있으며 화소당 8bit이면 256개의 회색명암준위를 얻는다. 세기정보를 기억시키는 또 하나의 방법은 매개 세기코드를 사용하는 출력장치에

서 직접 회색계조준위로 만드는 전압값으로 변환하는것이다.

여러가지 출력장치가 설치되여 사용할수 있는 경우에는 모든 현시장치들에서 동일한 색표이음부를 리용할수 있다. 이 경우 흑백현시장치에 대하여서는 색표가 그림 4-17 에서와 같이 RGB값의 범위를 리용하여 설정될수 있다. 주어 진 색첨수 ci에 대응하는 현시세기는

$$\text{intensity}(\text{세기}) = 0.5[\min(r, g, b) + \max(r, g, b)]$$

와 같이 계산된다.

표 4-2. 4준위회색계조체계에 대한 세기코드

세기코드	프레임완충기에 기억되는 세기코드 (2 진코드)		현시되는 회색계조
0.0	0	(00)	검은색
0.33	1	(01)	진한 회색
0.67	2	(10)	연한 회색
1.0	3	(11)	흰색

4절. 구역채우기속성

정의된 구역에 대한 채우기선택항목에는 옹근색채우기와 무늬채우기사이의 선택항목과 매개 색과 무늬에 대한 선택항목이 속한다. 이런 채우기선택항목들은 사용하는 프로그람의 능력에 따라 다각형구역 또는 곡선경계로 정의되는 구역에 적용될수 있다. 보충적으로 구역은 또한 여러가지 붓형태, 색, 투명도파라메터를 리용하여 그릴수도 있다.

채우기류형

구역은 세개의 기본적인 채우기류형 즉 색테두리가 있는 속이 빈 채우기, 옹근색채우기, 지적된 무늬 또는 도안채우기에 의하여 현시된다. 기본적인 채우기류형은 PHIGS프로그람에서 함수

```
setInteriorStyle(fs)
```

에 의하여 선택된다. 채우기류형파라메터 fs는 값 hollow(속이 빈), solid(옹근), pattern(무늬)을 가진다(그림 4-18). 채우기류형에 대한 다른 값으로서 hatch(격자)가 있는데 이것은 구역을 그림 4-19에서와 같이 선택된 격자무늬(평행선 또는 사귐선)로 채우는데 리용된다. 선의 속성에서와 마찬가지로 선택된 채우기류형값은 체계의 속성목록에 기록되며 후에 지적되는 구역의 내부를 채우는데 리용된다. 채우기류형파라메터 fs는 표준적으로 다각형구역에 적용되지만 그것들은 곡선경계를 가지는 구역채우기로도 실현시킬수 있다.

그림 4-18. 다각형채우기류형

그림 4-19. 살창무늬를 리용한 다각형채우기

속이 빈 구역은 경계륜곽선만을 리용하여 현시하며 내부색은 배경색과 같다. 옹근채우기는 구역의 테두리까지 포함하여 한가지 색으로 현시한다. 내부색 또는 속이 빈 구역의 륜곽선에 대한 색은

```
setInteriorColourIndex (fc)
```

에 의하여 선택한다.

여기서 채우기색파라메터 fc에는 요구되는 색코드가 설정된다. 다각형의 속이 빈 채우기는 선긋기루틴에 의하여 닫긴절선과 같이 발생시킨다. 구역의 옹근채우기는 3장 3절에서 설명한 주사선절차로 진행한다.

기타 채우기선택항목들로는 구역경계의 형, 경계너비, 경계색에 대한 지적이 들어 간다. 이 속성들은 채우기류형이나 채우기색과는 독립적으로 설정되며 선속성파라메터(선의 형, 선의 너비, 선의 색)와 같은 선택항목에 대하여 제공된다. 즉 내부를 어떻게 채우는가에는 무관계하게 구역의 경계를 점선 또는 파선, 굵게 또는 가늘게, 임의의 사용가능한 색으로 현시할수 있다.

무늬채우기

채우기무늬는

```
setInteriorStyleIndex (pi)
```

로 선택한다. 여기서 무늬첨수파라메터 pi는 표의 위치를 지적한다. 실례로 다음의 명령문모임은 fillArea명령에서 정의되는 구역을 무늬표에 기억된 두번째 무늬형으로 채우게 된다.

```
setInteriorStyle (pattern);
setInteriorStyleIndex (2);
fillArea (n, points);
```

격자무늬에 대하여서는 다른 표가 설치된다. 만일 이 프로그람토막에서 내부류형에 대하여 hatch 채우기를 선택하였다면 파라메터 pi에 할당되는 값은 격자무늬표에 기억된 무늬에 대한 첨수이다.

채우기류형 pattern에 대한 표의 항목값들은 개별적인 출력장치에서

```
setPatternRepresentation(ws, pi, nx, ny, cp)
```

에 의하여 만들어 지게 된다. 파라메터 pi는 워크스테이션 ws에 대한 무늬첨수번호이며 cp는 nx렬과 ny행을 가지는 색코드의 2차원배렬이다. 다음의 프로그람토막은 이 함수가 워크스테이션 1에 대하여 무늬표의 첫번째 항목값을 설정하는데 어떻게 리용되는가를 설명한다.

```
cp[1][1] := 4;   cp[2][2] := 4;
```

```
cp[1][2] := 0;   cp[2][1] := 0;
setPatternRepresentation(1, 1, 2, 2, cp);
```

표 4-3은 이 색표의 첫 두 항목값들을 보여 준다. 이 실례에서 색배렬 cp는 8색체계에서 붉은색과 검은색의 대각화소선들이 교차되는 무늬를 가리킨다.

색배렬 cp가 구역을 채우는데 적용될 때에는 배렬의 매 요소가 덮을 구역의 크기를 지적해야 한다. 이것은 무늬의 직4각형자리표범위를 설정하면 된다.

```
setPatternSize(dx, dy)
```

여기서 파라메터 dx와 dy는 배렬을 넘길 때의 자리표너비와 높이이다. 무늬배렬과 관련된 자리표크기의 실례를 그림 4-20에 주었다. 만일 이 그림에서 dx와 dy의 값이 화면자리표로 주어 진다면 색배렬의 매개 요소는 4개의 화소를 포함하는 2×2화면격자에 적용되게 된다.

pattern채우기를 시작하는 기준위치는 명령문

```
setPatternReferencePoint (position)
```

에 의하여 할당한다. 파라메터 position은 직4각형무늬의 왼쪽 아래구석에 놓이는 자리표 (xp, yp)에 대한 지시기이다. 그러면 이 시작위치로부터 무늬는 정의된 구역이 덮여 질 때까지 x와 y방향으로 무늬배렬이 겹치지 않게 복사된다. 직4각형무늬로 구역을 채우는 처리를 **타일붙이기**라고 부르며 직4각형채우기무늬를 때때로 **타일무늬**라고 한다. 그림 4-21에서는 무늬참조점으로부터 시작하여 3각형채우기구역의 타일붙이기를 보여 주었다.

표 4-3. 표 4-1 의 색코드를 리용하는 두개의 항목값을 가지는 워크스테이션무늬표

첨수(pi)	무늬(cp)
1	$\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$
2	$\begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}$

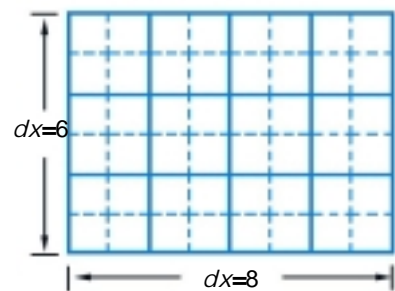



그림 4-20. 8×6 자리표 직4각형에 넘겨 진 4 개렬과 3개 행을 가지는 무늬배렬

무늬명령의 리용을 설명하기 위한 다음의 프로그람실례는 평행4변형채우기구역의 내부에 흑백무늬를 현시한다(그림 4-22). 이 프로그람에서 무늬의 크기는 매개 배렬요소가 하나의 화소에 넘어 가도록 설정하고 있다.

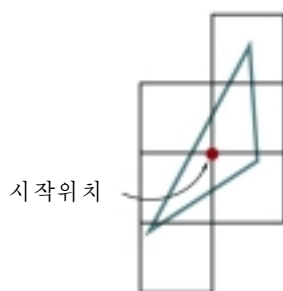



그림 4-21. 지적된 시작위치로부터 구역의 타일붙이기(정의된 구역을 통과하는 모든 주사선을 덮도록 무늬들이 겹치지 않게 린접되여 놓여 있다.)

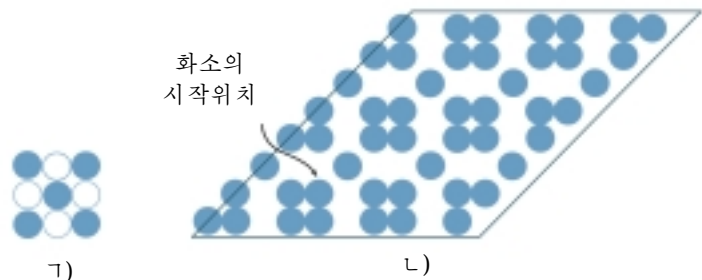



그림 4-22. 화면 ㄴ를 만들기 위하여 평행4변형채우기구역에 덧놓이는 무늬배렬 ㄱ

```
#define WS 1

void patternFill ()
{
  wcPt2 pts[4] ;
  int bwPattern[3] [3] ={1, 0, 0, 0, 1, 1, 1, 0, 0};

  pSetPatternRepresentation (WS, 8, 3, 3, bwPattern);

  pts[0].x = 10; pts[0].y = 10;
  pts[1].x = 20; pts[1].y = 10;
  pts[2].x = 28; pts[2].y = 18;
  pts[3].x = 18; pts[3].y = 18;

  pSetFillAreaInteriorStyle (PATTERN) ;
  pSetFillAreaPatternIndex (8) ;
  pSetPatternReferencePoint (14, 11);

  pFillArea (4, pts);
}
```

무늬채우기는 선택된 무늬가 주사선에 덧놓이도록 3장에서 설명한 주사선절차를 수정하여 실현할수 있다. 지적된 무늬채우기시작위치로부터 직4각형무늬를 수직으로는 채우기구역의 우와 아래경계주사선사이에, 수평으로는 이 주사선들우의 내부화소위치들에 사상한다. 수평으로 무늬배렬을 크기파라메터 dx의 값으로 지적되는 간격으로 반복한다. 류사하게 무늬의 수직반복은 파라메터 dy에 의하여 설정되는 간격으로 진행된다. 이 주사선무늬절차는 다각형과 곡선으로 경계 지어 지는 구역에도 다같이 적용된다.

격자채우기는 평행선모임을 현시하는 방법으로 구역에 적용한다. 채우기절차는 평행격자선 또는 격자선을 그려 실현한다. 격자선사이의 간격과 경사도는 격자무늬표안의 파라메터로 설정할수 있다. 라스터체계에서 격자채우기는 대각화소무리에 대하여 색값을 설정한 무늬배렬로 지적할수 있다.

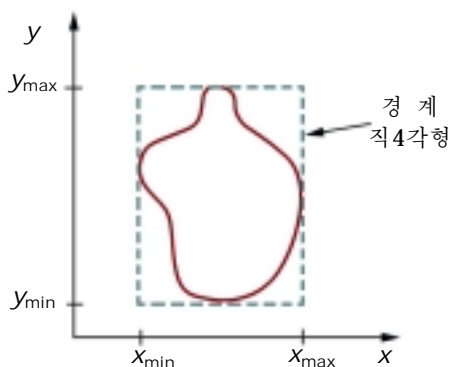



그림 4-23. x, y 방향에서 자리표범위 x_{min}, x_{max}, y_{min}, y_{max}가 가지는 구역에 대한 경계직4각형

많은 체계들에서 무늬참조점 (xp, yp)는 체계에서 준다. 실례로 참조점이 다각형의 정점에 자동적으로 설정될수 있다. 일반적으로는 참조점이 임의의 채우기구역의 자리표범위로 결정되는 경계직4각형(또는 경계덮개)의 왼쪽 아래구석으로 선택된다(그림 4-23). 참조자리표의 선택을 간단히 하기 위하여 일부 프로그람들은 화면자리표의 원점을 항상 무늬시작위치로 리용하며 창문체계들은 참조점을 흔히 창문의 자리표원점에 설정한다. (xp, yp)를 항상 자리표원점에 설정하면 무늬의 매개 색배렬요소가 하나의 화소에 사상될 때 타일붙이기조작이 간단해 진다. 실례로 무늬배렬에서 행위치가 거꾸로 참조된다면(즉 1에서 시작하여 밑에서부터 우로) 무늬값은 화면 또는 창문자리표로

화소위치 (x, y)에

```
setPixel(x, y, cp(y mod ny + 1, x mod nx + 1))
```

와 같이 할당된다. 여기서 ny 및 nx는 무늬배렬에서 행과 렬의 개수이다. 그러나 무늬시작위치를 자리표원점에 설정하면 무늬가 채우기구역보다 오히려 화면 또는 창문배경에 더 잘 부착되게 된다. 같은 무늬로 채워 지는 린접되여 있거나 또는 겹치는 구역들사이에는 명확한 경계가 나타나지 않게 된다. 또한 물체의 위치를 다시 정하고 동일한 무늬로 다시 채우면 물체내부에서 할당된 화소값들의 위치가 달라 질수 있다. 물체를 움직이면 고착된 내부무늬를 가지고 움직이지 않고 부동무늬의 배경이 살아 나게 된다.

채우기무늬는 또한 배경색(회색계조을 포함하여)과 여러가지 방법으로 결합시킬수 있다. 수자 1과 0만을 포함하는 비트배렬무늬에서 0값은 배경을 보여 주게 하는 투명지적자로 리용할수 있다. 또 다른 한가지는 1과 0의 수자들을 두가지 색무늬를 가지고 내부를 채우는데 리용할수 있다. 일반적으로 색채우기무늬는 배경색과 여러가지 방법으로 결합시킬수 있다. 무늬와 배경색은 론리연산을 리용하여 조합될수도 있고 배경색을 무늬색으로 간단히 바꿀수도 있다. 그림 4-24는 2값(흑백)체계에서 2×2채우기무늬에 대한 론리 및 치환조작이 매개 배경무늬화소값을 어떻게 설정하여 놓는가를 보여준다.

색배합채우기

채우기색이 배경색과 조합되도록 구역을 다시 그리는데 적용되는 수정된 경계채우기와 범람채우기절차를 **유연한채우기**(soft fill) 또는 **색배합채우기**(tint-fill)알고리듬이라고 한다. 이 채우기방법의 한가지 리용은 채우기색을 부드럽게 하여 물체의 터실터실한 경계를 깨끗하게 하는것이고 다른 하나의 리용은 본래 반투명붓으로 칠한 색구역을 다시 칠하는것이다. 이때 현재색은 붓의 색과 구역의 리면에 있는 배경색의 혼합이다. 어느 경우나 요구되는것은 구역우에서 현재의 채우기색과 새로운 채우기색이 같은 변화를 가지는것이다.

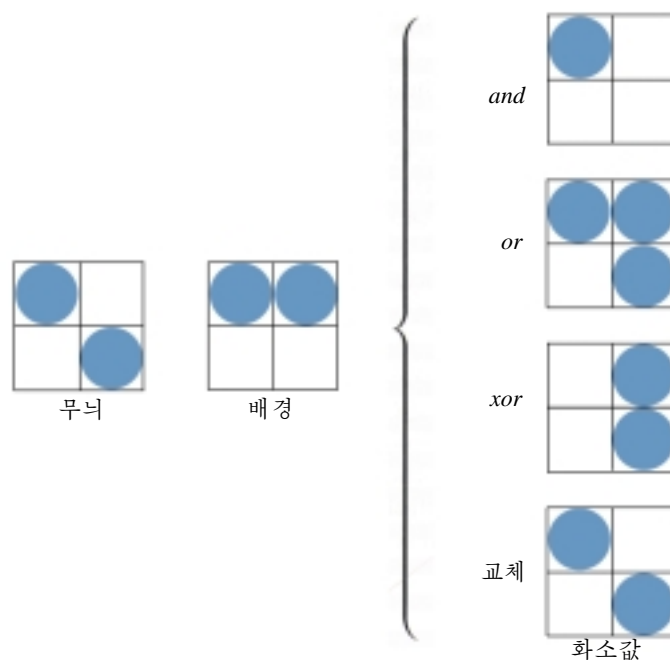



그림 4-24. 론리연산 and, or, xor(배타적 or)와 간단한 치환을 리용한 채우기무늬와 배경무늬의 조합

이런 형의 채우기의 실례인 선형적인 색배합채우기알고리듬에서는 본래 전경색 **F**와 단일한 배경색 **B**(**F**≠**B**)를 결합하여 그렸던 구역을 다시 그린다. **F**와 **B**의 값을 알고 있는 경우 프레임완충기의 현재의 색내용을 검사하면 이 색들이 본래 어떻게 결합되였는가를 결정할수 있다. 다시 채워 질 구역안의 매 화소의 현재의 RGB색 **P**는 **F**와 **B**의 선형결합이다.

$$\mathbf{P} = t\mathbf{F} + (1-t)\mathbf{B} \qquad (4-1)$$

여기서 투명도 t는 매 화소에 대하여 0∼1사이의 값을 가진다. 0, 5보다 작은 t값에서는 배경색이 채우기색보다 더 많이 구역의 내부색에 이바지한다. 벡토르방정식 4-1은 색의 매 RGB성분

$$\mathbf{P} = (p_R, p_G, p_B), \qquad \mathbf{F} = (F_R, F_G, F_B), \qquad \mathbf{B} = (B_R, B_G, B_B) \qquad (4-2)$$

들에 대하여 그대로 성립한다. 그러므로 파라메터 t의 값을 RGB색성분중 하나를 리용하여

$$t = \frac{P_k - B_k}{F_k - B_k} \qquad (4-3)$$

와 같이 계산할수 있다. 여기서 k=R, G, B이고 $F_k \neq B_k$ 이다. 리론적으로는 파라메터 t가 매 RGB성분에 대하여 같은 값을 가지지만 옹근수코드로의 자르기효과로 서로 다른 성분에서 t의 값이 서로 다를수 있다. 이 자르기오차는 **F**와 **B**사이에서 제일 큰 차를 가지는 성분을 선택함으로써 최소화할수 있다. t값은 후에 수정된 범람채우기 또는 경계채우기절차를 리용하여 새로운 채우기색 **NF**를 배경색과 섞는데 리용한다.

이러한 색배합채우기절차는 전경색을 바둑판무늬와 같은 다중배경색과 병합시키는 구역에도 적용할수 있다. 두가지 배경색 $\mathbf{B}_1$과 $\mathbf{B}_2$가 전경색 **F**와 혼합될 때 결과적인 화소색 **P**는

$$\mathbf{P} = t_0\mathbf{F} + t_1\mathbf{B}_1 + (1 - t_0 - t_1)\mathbf{B}_2 \qquad (4-4)$$

이다. 여기서 색항들의 결수 t_0, t_1, $(1-t_0-t_1)$의 합은 1과 같아야 한다. 두개의 비례파라메터 t_0과 t_1을 얻기 위하여 세개의 RGB색성분중 2개만을 리용하는 2개의 련립방정식을 설정할수 있다. 얻어진 파라메터들은 새로운 채우기색을 두 배경색과 혼합하여 새로운 화소색을 얻는데 리용한다. 세개의 배경색과 한개의 전경색, 2개의 배경색과 2개의 전경색에 대하여서는 4가지 색의 비례를 얻기 위하여 각각 3개의 RGB방정식이 필요하다. 그러나 전경색과 배경색의 일부 조합에서는 2 또는 3개 련립방정식이 풀리지 않을수 있다. 이것은 색값들이 모두 대단히 류사하거나 또는 서로 비례할 때 일어 난다.

5절. 문자의 속성

현시되는 문자의 모양은 폰트, 크기, 색, 방향과 같은 속성에 의하여 조종된다. 속성은 전체 문자렬(본문) 및 표식기호와 같이 정의되는 개별문자들에 다같이 설정될수 있다.

본문속성

도형처리프로그람작성자들이 사용할수 있는 본문선택항목들은 대단히 많다. 무엇보다먼저 폰트선택이다. 여기서 폰트는 New York, Courier, Helvetica, London, Times Roman 과 같은 특정한 설계형태를 가지는 문자모임과 여러가지 특수기호모임을 가리킨다. 선택된 폰트의 문자들은 아래밑선(실선, 점선, 2 중선)과 결합된 류형 또는 **굵은체(boldface)**, *경사체(italics)*, 륜곽체(outline), 그림자체(shadow) 류형으로도 현시될수 있다.

PHIGS프로그람에서 매개 폰트 및 그와 관련된 류형은 함수

```
setTextFont (tf)
```

의 본문폰트파라메터 tf에 옹근수코드를 설정하여 선택한다. 폰트선택항목은 미리 정의된 격자무늬모임 또는 절선과 스플라인곡선으로 설계되는 문자모임을 사용할수 있게 한다.

현시되는 본문에 대한 색설정은 체계속성목록에 기억되며 문자의 정의를 프레임완충기에 적재하는 절차에 리용된다. 문자렬이 현시될 때 현재색이 프레임완충기에서 문자형태 및 위치에 대응하는 화소값들을 설정하는데 리용된다.

본문색(또는 세기)은 응용프로그람에서

```
setTextColourIndex (tc)
```

에 의하여 조종된다. 여기서 본문색파라메터 tc는 허용되는 색코드를 가리킨다.

본문의 크기는 문자의 총체적인 크기(높이 및 너비)를 비례변환하거나 또는 너비만을 비례변환하여 조정할수 있다. 인쇄기나 식자공은 문자의 크기를 포인트(point)로 지적한다. 1포인트는 0.013837in(약 0.353mm)이다. 실례로 이 책 원문은 10.5포인트 폰트이다. 포인트값은 문자본체의 크기(그림 4-25)를 지적하지만 동일한 포인트로 지적되는 폰트라도 서체의 설계에 따라 서로 다른 문자크기를 가질수 있다. 문자본체의 밑선과 웃선사이의 거리는 정해 진 크기와 서체에서 모든 문자에 대하여 동일하지만 본체의 너비는 다를수 있다. 비례간격폰트는 i, j, l, f와 같은 좁은 문자에는 W, M과 같은 넓은 문자에 비하여 보다 작은 본체너비를 할당한다. 문자의 높이는 문자의 기준선과 모자선사이의 거리로 정의한다. 그림 4-25에서 f 및 j와 같은 장식꼬리문자는 일반적으로 문자본체한계를 넘어서 확장되며 하행이 있는 글자(g, j, p, q, y)는 기준선아래로 확장된다. 폰트설계자는 문자본체안에서 매개 문자에 대하여 본문이 문자본체들이 접촉되면서 현시될 때 인쇄행을 따라서 그사이에 적당한 간격이 생기도록 위치를 정한다.

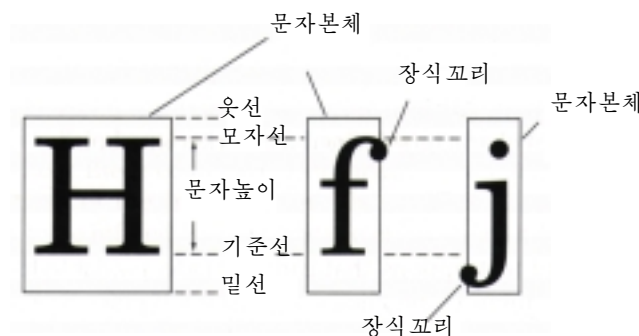



그림 4-25. 문자본체

본문의 크기는 문자의 높이에 대한 너비의 비를 변화시킴이 없이

```
setCharacterHeight(ch)
```

에 의하여 조종할수 있다. 파라메터 ch에는 대문자의 자리표높이 즉 사용자자리표로 기준선과 모자선사이거리를 설정하는 0보다 큰 실수값이 할당된다. 이 설정은 또한 문자본체크기에 영향을 미치며 그리하여 문자의 너비와 간격은 동일한 본문비례를 유지하도록 조정된다. 실례로 높이를 2배하면 또한 문자의 너비와 문자사이간격도 2배 된다. 그림 4-26에서는 세가지 서로 다른 문자높이로 현시된 문자렬을 보여 주었다.

그림 4-26. 본문현시에서 서로 다른 문자높이의 설정효과

본문의 너비는 함수

```
setCharacterExpansionFactor (cw)
```

에 의하여 설정할수 있다. 여기서 문자너비파라메터 cw에는 문자의 본체너비를 비례변환하는 정의실수값이 설정된다. 이 속성설정은 본문의 높이에는 영향을 미치지 않는다. 서로 다른 문자너비로 현시되는 본문의 실례를 그림 4-27에 보여 주었다.

문자사이간격은

```
setCharacterSpacing (cs)
```

에 의하여 따로 조종한다. 여기서 문자간격파라메터 cs에는 임의의 실수값이 할당될수 있다. cs에 할당된 값은 인쇄행을 따르는 문자본체들사이의 간격을 결정한다. cs에 대한 부의 값은 문자본체들을 겹치게 하고 정의값은 현시되는 문자들을 벌려 놓기 위하여 공백을 삽입한다. cs에 값 0을 할당하면 문자본체들사이에 공백이 없이 본문이 현시되게 한다. 공백크기는 cs의 값을 문자높이(기준선과 모자선사이거리)에 곱하여 결정한다. 그림 4-28의 문자렬은 문자간격파라메터에 대한 세가지 서로 다른 설정으로 현시된것이다.

그림 4-27. 본문현시에서 서로 다른 문자너비의 설정효과

그림 4-28. 본문현시에서 서로 다른 문자간격의 효과

현시되는 문자렬의 방향은 문자웃벡토르(UpVector)의 방향에 따라 설정된다.

```
setCharacterUpVector (upvect)
```

이 함수에서 파라메터 upvect에는 벡토르의 *x*와 *y*성분을 가리키는 두개의 값이 할당된다. 이때 본문은 기준선으로부터 모자선으로 향하는 문자방향이 웃벡토르방향이 되도록 현시된다. 실례로 upvect=(1, 1)인 경우 웃벡토르의 방향은 45°이며 본문은 그림 4-29에서 보여 주는것처럼 현시되게 된

다. 본문방향설정절차는 문자들을 기준선으로부터 모자선으로 향하는 문자본체의 측면이 웃벡토르와 일직선이 되도록 회전시킨다. 회전된 문자모양은 그다음 프레임완충기에 주사변환된다.

문자렬을 수직 또는 수평으로 배렬할수 있으면 많은 응용들에서 쓸모 있다(그림 4-30). 이 선택항목에 대한 속성파라메터는

```
setTextPath (tp)
```

명령문에 의하여 설정된다. 여기서 본문경로파라메터 tp에는 값 *right*(오른쪽), *left*(왼쪽), *up*(우), *down*(아래)이 할당될수 있다. 이 4가지 선택항목에 의하여 현시되는 본문의 실례를 그림 4-31에 보여 주었다. 이 선택항목을 실현하는 절차는 문자렬무늬를 프레임완충기에 넘기기전에 지적된 방향으로 변환하여야 한다.

그림 4-29. 웃벡토르의 방향(ㄱ)은 현시되는 본문의 방향(ㄴ)을 조종한다.

문자렬은 또한 경사진 본문을 만들기 위하여 웃벡토르와 본문경로지적을 결합하여 방향을 지적할수 있다. 그림 4-32에서는 45° 웃벡토르에 여러가지 본문경로를 설정하여 발생시킨 문자렬의 방향을 보여 주었다. 이 웃벡토르에 본문경로값 *down*과 *right*를 주어 발생시킨 본문의 실례를 그림 4-33에 보여 주었다.

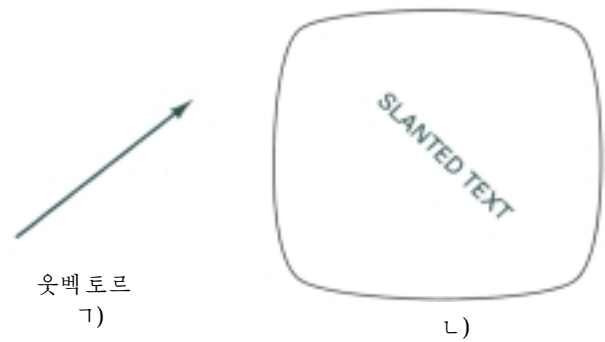
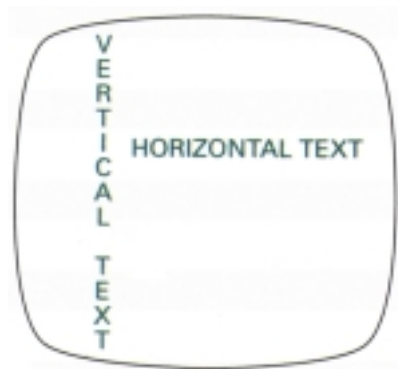



그림 4-30. 본문경로속성은 문자렬의 수평 또는 수직배렬을 만들기 위하여 설정된다.

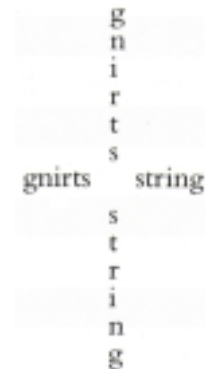



그림 4-31. 4가지 본문경로선택항목으로 현시된 본문

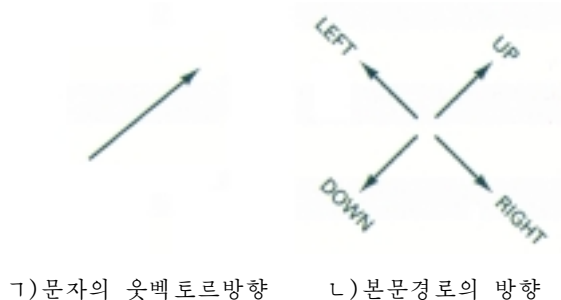



그림 4-32. 웃벡토르의 지적(ㄱ)은 본문경로의 방향(ㄴ)을 조종한다.

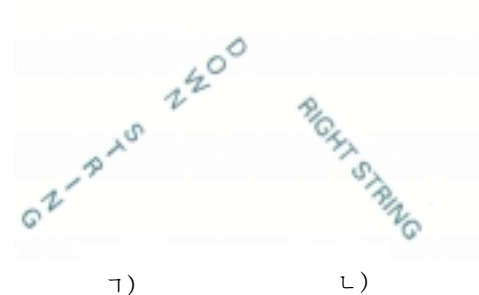



그림 4-33. 그림 4-32에서 45° 웃벡토르는 *down*경로에 대하여서는 화면 ㄱ를, *right*경로에 대하여서는 화면 ㄴ를 만든다.

문자렬에 대한 다른 편리한 속성은 기준맞추기이다. 이 속성은 본문이 시작자리표에 대하여 어떻게 위치를 정하는가를 지적한다.

기준맞추기속성은

```
setTextAlignment (h, v)
```

로 설정한다. 여기서 파라메터 h와 v는 각각 수평 및 수직기준맞추기를 조종한다. 수평기준맞추기는 h에 값 *left*(왼쪽), *center*(중간), *right*(오른쪽)를 할당하여 설정한다. 수직기준맞추기는 v에 값 *top*(웃선), *cap*(모자선), *half*(절반선), *base*(기준선), *bottom*(밑선)을 할당하여 설정한다. 이 기준맞추기값들에 대한 해석은 본문경로에 대한 현재설정에 관계된다. 그림 4-34에서는 본문이 수평오른쪽으로 또는 수직아래로 현시될 때 기준맞추기설정의 위치를 보여 주었다. 이러한 해석은 본문경로값 *left* 및 *up*에도 적용된다. 매개 본문경로에 대한 제일 자연적인 기준맞추기는 h와 v파라메터에 값 *normal*을 할당하는것이다. 그림 4-35는 수평 및 수직본문표식에 대한 일반적인 기준맞추기위치를 설명하고 있다.

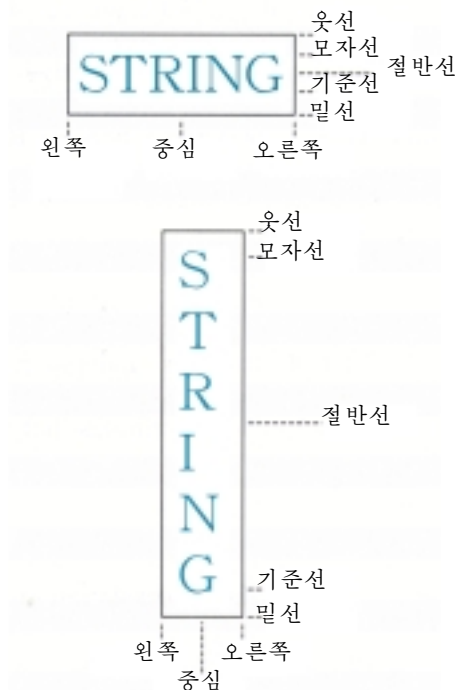



그림 4-34. 수평 및 수직문자렬에 대한 기준맞추기속성값

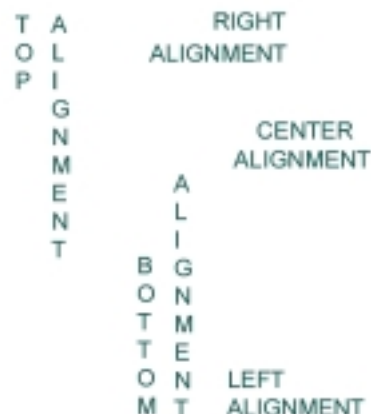



그림 4-35. 문자렬기준맞추기

본문현시에 대한 정밀도는

```
setTextPrecision (tpr)
```

에 의하여 지적된다. 여기서 본문정밀도파라메터 tpr에는 값 *string*, *char*, *stroke*중 하나가 할당된다.

고급한 본문은 정밀도파라메터가 값 *stroke*로 설정될 때 현시된다. 이 정밀도로 설정되면 문자형태를 정의할 때 보다 자세한 세부가 리용되며 속성선택의 처리와 기타 문자렬처리절차들이 제일 높은 정밀도로 진행되게 된다. 제일 낮은 급의 정밀도설정 *string*은 문자렬의 보다 빠른 현시에 리용된다. 이 정밀도에서는 본문경로와 같은 많은 속성선택들이 무시되며 문자렬처리절차들이 간단해 져서 처리시간이 줄어 들게 된다.

표식속성

표식기호는 서로 다른 색과 서로 다른 크기로 현시할수 있는 하나의 문자이다. 표식속성은 선택된 문자를 정해 진 위치에 지적된 색과 크기로 라스터에 적재하는 절차에 의하여 실현된다.

표식기호로 되는 매개 문자는

```
setMarkerType (mt)
```

에 의하여 선택된다. 여기서 표식의 형파라메터 mt는 옹근수코드로 설정된다. 표식의 형에 대한 대표적인 코드는 각각 점(·), 더하기표식(+), 별표(*), 동그라미(○), 곱하기표식(×)을 나타내는 1~5사이의 옹근수이다. 현시되는 표식은 자리표에 중심이 맞추어 진다.

표식의 크기는

```
setMarkerSizeScaleFactor (ms)
```

로 설정한다. 여기서 표식의 크기파라메터 ms에는 정의 수가 할당된다. 이 비례변환파라메터는 선택된 매 표식기호의 표준크기에 적용된다. 1보다 큰 값은 문자를 확대시키고 1보다 작은 값은 표식의 크기를 줄인다.

표식의 색은

```
setPolymarkerColourIndex (mc)
```

에 의하여 지적된다. 파라메터 mc에 선택된 색코드는 현재속성목록에 기억되며 후에 지적되는 표식기초요소를 현시하는데 리용된다.

6절. 묶음속성

지금까지 고찰한 절차들에서 매개 함수는 기초요소들이 속성설정에 의하여 어떻게 현시되는가를 정확히 지적하는 한가지 속성만을 가지고 있다. 이런 지적을 개별(또는 묶지 않은)속성이라고 하는데 이것은 기초요소들을 지적된 방식으로 현시할수 있는 출력장치에 리용되게 된다. 개별속성을 리용하는 응용프로그람이 여러가지 출력장치와 결합되는 경우 일부 장치들이 의도하는 속성을 현시할 능력이 없을수 있다. 실례로 개별적인 색속성을 리용하는 응용프로그람을 흑백색현시장치에서 쓰려면 그것을 흑백현시장치에서 받아 들일수 있게 수정하여야 한다.

개별속성명령은 하나의 출력장치가 리용될 때는 속성지적이 간단하고 직접적인 방법이다. 여러가지 종류의 출력장치가 설치되여 사용될수 있는 경우에는 사용자에게 속성들이 서로 다른 매 장치에서 어떻게 해석되는가를 사용자가 지적할수 있으면 편리하다. 이것은 매 출력장치들에 대하여 매개 기초요소들의 현시에 리용할 속성값들의 모임을 기입하는 표를 설정하면 된다. 이때 각 출력장치에서 쓸 기초요소들에 대한 속성값들의 모임은 해당한 표첨수를 지적하여 선택한다. 이 방식으로 지적되는 속성들을 **묶음속성**(bundled attributes)이라고 한다. 매개 기초요소를 개별적인 출력장치에서 현시할 때 리용할 속성값들의 모임을 정의하는 표를 **묶음표**(bundle table)라고 한다.

워크스테이션표항목값들로 묶을수 있는 속성들은 색 및 선형태와 같이 자리표지적을 포함하지 않는것들이다. 묶은 또는 묶지 않은 지적사이의 선택은 이 매개 속성들에 대하여 관점원기발(aspect

source flag)이라고 하는 스위치를 설정하여 진행한다.

```
setIndividualASF (attributeptr, flagptr)
```

여기서 파라메터 attributeptr는 속성들의 목록을 가리키며 파라메터 flagptr는 대응하는 관점원기발들의 목록을 가리킨다. 관점원기발에는 값 individual(개별) 또는 bundled(묶음)가 할당될수 있다. 묶을수 있는 속성들을 다음 부분에서 설명한다.

직선의 묶음속성

지적된 워크스테이션에서 선속성에 대한 묶음표의 항목값들은 함수

```
setPolylineRepresentation (ws, li, lt, lw, lc)
```

에 의하여 설정한다. 파라메터 ws는 워크스테이션식별자이며 선첨수파라메터 li는 묶음표의 위치를 규정한다. 이때 파라메터 lt, lw, lc들은 묶어 지며 지적된 표첨수에 대하여 각각 선형태, 선너비, 선색을 설정하는 값들이 할당된다. 실례로 다음의 명령문들은 두개의 서로 다른 워크스테이션에서 첨수번호 3으로 지적되는 선속성들의 묶음을 정의한다.

```
setPolylineRepresentation (1, 3, 2, 0.5, 1);
setPolylineRepresentation (4, 3, 1, 1, 7);
```

이때 표첨수값 3이 할당되는 절선은 워크스테이션 1에서는 파선, 절반두께, 푸른색을 리용하여 현시되게 되며 워크스테이션 4에서는 동일한 첨수가 실선, 표준크기, 흰색선으로 그려 진다.

일단 묶음표가 설정되면 선의 묶음속성모임은 매개 워크스테이션에 대하여 표첨수값을 지적하여 선택한다.

```
SetPolylineIndex (li)
```

이후의 polyline명령들은 워크스테이션에서 선첨수파라메터 li값에 의하여 지적되는 표위치에서 정의되는 묶음속성값들의 모임을 가지고 선을 발생시킨다.

구역채우기의 묶음속성

구역채우기의 묶음속성에 대한 표의 항목값들은

```
setInteriorRepresentation (ws, fi, fs, pi, fc)
```

에 의하여 설정된다. 이것은 워크스테이션 ws의 채우기첨수 fi에 대응하는 속성목록을 정의한다. 파라메터 fs, pi, fc에는 각각 지적된 워크스테이션의 채우기류형, 무늬첨수, 채우기색에 대한 값들이 할당된다. 이러한 묶음표는 또한 다각형채우기구역의 경계속성에 대하여서도 설정할수 있다.

그러면 매개 속성모임은 함수

```
setInteriorIndex (fi)
```

에 의하여 표에서 선택된다. 후에 정의되는 채우기구역들은 매 능동워크스테이션에서 채우기첨수파라

메터 fi로 지적되는 표의 항목값에 따라 현시된다. 무늬참조점 및 무늬크기와 같은 구역채우기의 다른 속성들은 워크스테이션지적과는 관계없으며 앞에서 서술된 함수들에 의하여 설정한다.

본문의 묶음속성

함수

```
setTextRepresentation (ws, ti, tf, tp, te, ts, tc)
```

는 본문의 폰트, 정밀도, 비례결수, 크기, 색에 대한 값들을 워크스테이션 ws에 대하여 본문첨수파라메터 ti에 할당되는 값에 의하여 지적되는 표위치에 묶는다. 문자웃벡토르, 본문경로, 문자높이, 본문기준맞추기를 비롯한 본문의 다른 속성들은 개별적으로 설정한다.

매개 본문첨수값은 함수

```
setTextIndex (ti)
```

에 의하여 선택한다. 그러면 호출되는 매개 본문함수는 매 워크스테이션에서 이 표위치로 지적되는 속성들의 모임으로 현시된다.

표식의 묶음속성

표식의 묶음속성에 대한 표의 항목값들은

```
setPolymarkerRepresentation (ws, mi, mt, ms, mc)
```

에 의하여 설정한다. 이것은 워크스테이션 ws에서 첨수 mi에 대한 표식의 형, 비례결수, 색을 정의한다. 묶음표는 함수

```
setPolymarkerIndex (mi)
```

에 의하여 선택된다.

7절. 질문함수

체계목록에서 워크스테이션의 형 및 상태와 같은 속성 기타 다른 파라메터들에 대한 현재의 설정은 질문함수에 의하여 꺼내볼수 있다. 이 함수들은 지적된 파라메터에 현재값들을 복사한다. 이 값들은 후에 다시 리용하기 위하여 보관하거나 또는 오유가 발생하였을 때 체계의 현재상태를 검사하는데 리용할수 있다.

현재속성값은 질문함수에서 속성의 이름을 지적하여 검사한다. 실례로 함수

```
inquirePolylineIndex (lastli)
```

와

```
inquireInteriorColorIndex (lastfc)
```

는 선첨수 및 채우기색에 대한 현재값을 파라메터 lastli와 lastfc에 복사한다.다음의 프로그람토막은 선모임이 새로운 선형태로 그려 진후 현재의 선형태값을 다시 리용하는것을 보여 주고 있다.

```
inquireLinetype (oldlt)
setLinetype (newlt)
      .
      .
      .
setLinetype (oldlt)
```

8절. 경계허상제거

3장에서 설명된 라스터알고리듬에 의하여 만들어 져 현시되는 기초요소들은 터실터실하거나 또는 계단모양을 띠게 되는데 이것은 표본화처리에 의하여 물체의 자리표점들이 불련속적인 옹근수화소위치로 수자화되기때문이다. 낮은주파수표본화(거친표본화)로 인한 정보의 외곡을 **경계허상**(aliasing)이라고 한다. 현시되는 라스터선의 모양은 거친표본화처리를 보상하는 **경계허상제거**(antialiasing)방법을 적용하여 개선할수 있다.

그림 4-36에 거친표본화결과의 실례를 보여 주었다. 이런 주기적인 물체에서 정보의 손실을 피하기 위하여서는 표본화주파수를 최소한 물체에서 일어 나는 제일 높은 주파수의 2배(나이크위스트표본화주파수 또는 속도라고 한다.)로 설정하여야 한다.

$$f_s = 2f_{max} \tag{4-5}$$

달리 말하면 표본화간격은 주기간격의 절반(나이크위스트표본화간격이라고 한다.)보다 크지 말아야 한다는것이다. x축표본화에서 나이크위스트표본화간격 Δx_s는

$$\Delta x_s = \frac{\Delta x_{cycle}}{2} \tag{4-6}$$

이다. 여기서 $\Delta x_{cycle}=1/f_{max}$이다. 그림 4-36에서 표본화간격은 주기간격의 1½배이다. 즉 표본화간격이 나이크위스트표본화간격의 3배나 된다. 이 실례에서 물체의 모든 정보를 회복시키려고 한다면 표본화간격을 그림에 보여 준 크기의 1/3이하로 하여야 한다.

라스터체계에서 표본화속도를 높이는 한가지 방법은 간단히 물체를 보다 높은 분해능으로 현시하는것이다. 그러나 설사 현재기술로 가능한 제일 높은 분해능이라고 해도 터실터실한것이 어느 정도 보인다. 프레임완충기를 크게 할수 있는데는 제한이 있으며 의연히 재생속도는 초당 30~60프레임으로 유지한다.

그림 4-36. ㄱ의 주기적인 형태를 *로 표식된 위치들에서 표본화하는것은 ㄴ의 낮은주파수의 외곡된 표현을 가져 온다.

그리고 물체를 련속적인 파라메터들로 정확하게 표현하기 위해서는 매우 작은 표본화간격이 필요하다. 따라서 하드웨어기술이 아주 큰 프레임완충기를 조종할수 있게 되여 있지 않는 한 화면분해능을 높이는것은 경계허상문제의 완전한 해결책이 못된다.

둘이상의 세기준위(색 또는 회색계조)를 현시할수 있는 라스터체계에서는 화소의 세기를 수정하는 경계허상제거방법을 적용할수 있다. 기초요소들의 경계를 따라 화소의 세기들을 적당히 변화시키면 터실터실한것들이 작게 나타나도록 경계를 원활하게 할

수 있다. 간단한 경계허상제거방법은 화면을 마치 실제로 사용가능한것보다 더 미세한 격자로 덮여진것처럼 보고 표본화주파수를 높이는것이다. 다음에 이 미세한 격자를 지나가는 여러 표본점들을 리용하여 화면의 매 화소에 대하여 해당한 세기준위를 결정할수 있다. 물체의 특성을 보다 높은 분해능으로 표본화하고 결과를 낮은 분해능으로 현시하는 이 기술을 초표본화(supersampling) 또는 후려파법(postfiltering, 왜냐하면 일반적으로 부분화소격자위치에서 세기를 계산하고 다음에 결과를 화소세기를 얻기 위하여 결합하기때문이다.)이라고 한다. 현시되는 화소위치는 매우 작은 수학적인 점이 아니라 화면의 유한한 구역을 덮는 빛점이다. 지금까지 설명한 선 및 구역채우기알고리듬들에서는 매 화소의 세기를 물체경계의 하나의 점위치에 의하여 결정하였다. 초표본화를 진행하면 화소의 세기정보를 화소의 총체적인 세기에 이바지하는 여러점들로부터 얻게 된다.

초표본화의 다른 한가지 방법은 매개 화소에서 현시되는 물체와 겹치는 구역을 계산하여 화소의 세기를 결정하는것이다. 겹침구역계산에 의한 경계허상제거방법을 구역표본화(area sampling) 또는 선려파법(prefiltering, 왜냐하면 전체로서의 화소의 세기가 부분화소세기를 계산함이 없이 결정되기때문이다.)이라고 한다. 화소의 겹침구역은 물체의 경계가 개별적인 화소구역의 어디로 지나가는가를 결정하면 얻어 진다.

라스터물체들은 또한 화소구역의 현시위치를 밀기하면 경계허상이 제거될수 있다. 화소위치조정(pixel phasing)이라고 하는 이 방법은 전자속을 물체외형에 대하여 《미크로위치설정》하게 한다.

선분의 초표본화

직선의 초표본화는 여러가지 방법으로 진행할수 있다. 회색계조화면인 경우에는 선분에서 매개 화소를 몇개의 부분화소들로 나누어 놓고 선경로를 따르는 부분화소들의 개수를 셀수 있다. 그다음 매 화소에 대한 세기준위를 이 부분화소들의 개수에 비례하는 값으로 설정한다. 이 방법의 실례를 그림 4-37에 주었다. 매개 바른4각형화소구역을 9개의 동일한 크기의 바른4각형부분화소들로 분할하였다. 색칠한 구역들은 브리센함의 알고리듬에 의하여 선택될수 있는 부분화소들을 나타낸다. 이 방법은 임의의 화소안에서 선택될수 있는 부분화소의 최대수가 3이기때문에 0우로 세가지 세기값을 설정할수 있게 한다. 이 실례에서 (10, 20)위치의 화소는 최대세기(준위3)로 설정되고 (11, 21)과 (12, 21)의 화소들은 두번째로 높은 세기(준위2)로 설정되며 (11, 20), (12, 22)의 화소들은 0우의 제일 낮은 세기(준위1)로 설정된다. 그러므로 선의 세기는 많은 화소들우에 펼쳐 지며 계단효과는 계단걸음의 근방(수평행정사이)에서 약간 흐려 진 선경로를 현시함으로써 원활하게 된다. 이 방법으로 선의 경계허상을 제거하는데 더 많은 세기준위를 리용하려면 매 화소를 가로지르는 표본화위치들의 수를 증가시키면 된다. 16개의 부분화소들은 0우에 4개의 세기준위를 주며 25개의 부분화소들은 5개의 준위를 준다.

그림 4-37. 왼쪽끝점이 화면자리표 (10,20)에 있는 선분을 초표본화하는 부분화소들의 위치

그림 4-37의 초표본화실례에서는 유한크기의 화소구역을 고찰하였지만 선의 너비는 0인 수학적인 존재로 취급하였다. 실제로 현시되는 선은 화소와 거의 같은 너비를 가진다. 선의 유한너비를 고려한다면 매개 화소에 대하여 선구역을 나타내는 다각형내부에 들어 가는 부분화소들의 개수에 비례하는 세기를 설정하는 방법으로 초표본화를 진행할수 있다. 부분화소들은 그의 왼쪽 아래구석이 다각형경계안에 있으면 선내부에 있다고 본다. 이 표본화절차의 우점은 매 화소에 대하여 줄수 있는 세기준위의 개수가 화소구역안의 부분화소들의 총수와 같다는것이다. 그림 4-37의 실례에

대하여 말한다면 그림 4-38에서와 같이 선경로에 평행인 다각형경계들의 위치를 지정함으로써 유한너비를 가지는 선으로 표현할수 있다. 매개 화소들은 이때 0우에서 가능한 9개의 밝기준위중 어느 하나로 설정될수 있다.

유한너비선에 의한 초표본화의 다른 우점은 선의 총세기가 더 많은 화소들에 분포된다는것이다. 그림 4-38에서는 격자위치 (10, 21)의 화소가 켜지며(세기준위 2) (10, 21)위치의 아래와 오른쪽 화소들로부터 지원을 받는다. 또한 색현시장치인 경우에는 배경색도 고려하도록 방법을 확장할수 있다. 어떤 선들은 서로 다른 여러 색구역을 지나갈수 있다. 이때에는 화소의 색을 설정하기 위하여 부분화소들의 세기를 평균할수 있다. 실례로 어떤 화소구역안의 5개 부분화소가 붉은색선의 경계안에 있고 남은 4개는 푸른색배경구역안에 떨어 진다면 이 화소에 대한 색을

$$\mathrm{pixel}_{\mathrm{color}} = (5 \cdot \mathrm{red} + 4 \cdot \mathrm{blue})/9$$

와 같이 계산할수 있다.

유한너비선초표본화는 이러한 우점이 있는 반면에 내부부분화소의 식별에서 간단히 부분화소가 선경로를 따르는가만 결정할 때보다 더 많은 계산을 요구하는 부족점이 있다. 이 계산들은 또한 선경로에 대한 선경계들의 위치를 지정해야 하므로 복잡해 진다. 이 위치선정은 선의 경사도에 관계된다. 45°선에 대한 선경로는 다각형구역의 중심을 지난다. 그러나 수평 또는 수직선에 대하여서는 선경로가 다각형경계들중의 하나로 되여야 한다. 실례로 격자자리표 (10, 20)을 통과하는 수평선은 수평격자선 y=20과 y=21에 의하여 경계 지어 지는 다각형으로 표현될수 있다. 류사하게 (10, 20)을 통과하는 수직선을 표현하는 다각형은 격자선 x=10과 x=11을 따르는 수직경계들을 가질수 있다. 경사도 $|m|<1$인 선에 대한 수학적인 선경로는 상대적으로 다각형의 아래경계에 더 가깝게 놓이며 경사도 $|m|>1$인 선에서는 다각형의 웃경계에 더 가깝게 놓인다.

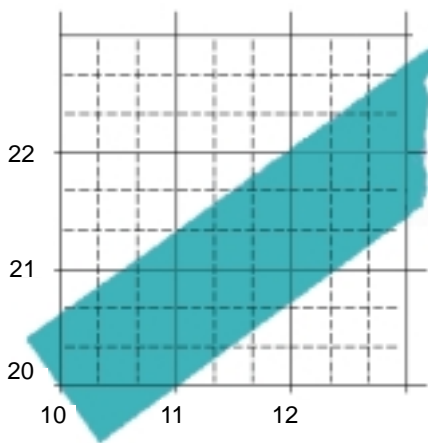



그림 4-38. 유한너비선 내부에서의 초표본화부분화소위치들

무게 붙은 화소마스크

초표본화알고리듬은 흔히 화소구역의 중심에 가까운 부분화소들에 더 많은 무게를 주어 실현한다. 왜냐하면 이 부분화소들이 화소의 총체적인 세기결정에서 더 중요하다고 볼수 있기때문이다. 이제까지 고찰한 3×3화소부분분할에서는 그림 4-39에서와 같은 무게붙이기방안을 리용할수 있다. 여기서 중심부분화소는 무게가 구석부분화소들의 4배, 나머지 부분화소들의 2배이다. 이때 9개 부분화소들의 매 격자에 대하여 계산되는 세기는 중심부분화소는 1/4무게결수를 가지며 우, 아래 그리고 측면의 부분화소들은 1/8무게결수를 가지며 구석부분화소들은 1/16무게결수를 가진다. 부분화소들의 상대적중요성을 지적하는 값들의 배렬을 흔히 부분화소무게들의 《마스크》라고 한다. 이러한 마스크는 더 큰 부분화소격자에도 설정할수 있다. 또한 이런 마스크는 흔히 린접화소에 속하는 부분화소들이 기여하는 몫도 들어 가도록 확장되며 따라서 세기는 린접화소들우에서 평균될수 있다.

1	2	1
2	4	2
1	2	1

그림 4-39. 3×3부분화소살창에 대한 상대적무게값

선분의 구역표본화

직선의 구역표본화는 매 화소에 대하여 그것이 유한너비선과 겹치는 부분에 비례하는 세기를 설정하는 방법으로 진행한다. 선은 직4각형처럼 취급할수 있으며 따라서 두개의 린접한 수직(또는 두개의 린접한 수평)화면격자선사이의 직선의 부분구역은 제형으로 된다. 화소겹침구역은 이 제형들이 수직렬(또는 수평형)의 매 화소를 얼마나 많이 덮는가를 결정하여 계산한다. 그림 4-38에서 화면격자자리표가 (10, 20)인 화소는 선구역에 의해 약 90% 덮여 지며 따라서 그의 세기는 최대세기의 90%로 설정할수 있다. 류사하게 (10, 21)의 화소는 최대값의 약 15%세기로 설정할수 있다. 화소의 겹침구역을 평가하는 방법은 그림 4-38의 초표본화실례에 의하여 설명된다. 선경계안의 부분화소의 총수는 대략 겹침구역과 같다. 이 평가값은 부분화소격자를 더 미세하게 하면 개선된다. 색현시장치인 경우에는 다른 색구역들과의 화소의 겹침구역들을 계산하고 최종적인 화소색은 여러 겹침구역들의 평균색으로 취한다.

려파기술

선의 경계허상을 제거하는 더 정확한 방법은 려파기술을 리용하는것이다. 이 방법은 무게 붙은 화소마스크를 적용하는것과 류사하지만 여기서는 화소를 덮는 련속무게면(또는 려파함수)을 상상한다. 그림 4-40에 직4각형, 원추, 가우스려파함수의 실례를 보여 주었다. 려파함수를 적용하는 방법은 무게 붙은 마스크를 적용하는것과 류사하지만 여기서는 무게 붙은 평균세기를 얻기 위하여 화소면우에서 적분한다. 일반적으로 적분계산에서 계산을 줄이기 위하여 검색표를 리용한다.

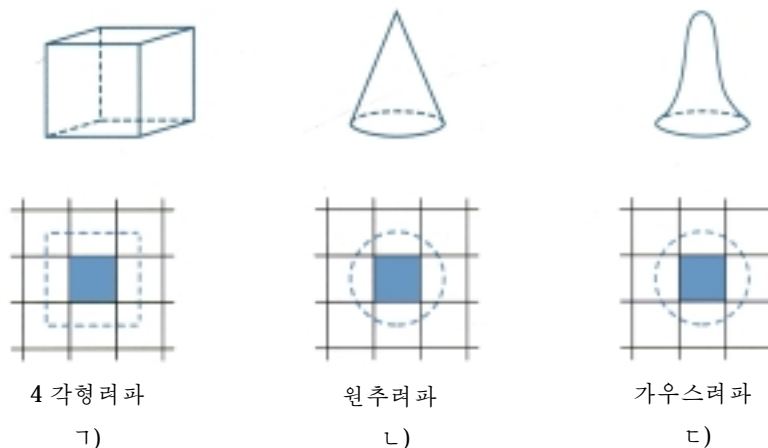



그림 4-40. 선경로의 경계허상을 제거하는데 리용되는 일반적인 려파함수(매개 려파기의 체적은 1로 정규화하며 높이는 임의의 부분화소위치에서의 상대무게를 준다.)

화소위치조정

부분화소위치를 화면격자안에서 주소화할수 있는 라스터체계에서는 화소의 위치조정을 물체의 경계허상제거에 써먹을수 있다. 선경로 또는 물체경계의 계단모양은 전자속을 물체형태에 의하여 지적되는 위치들에 더 가깝게 근사화되도록 움직이면(미크로위치지정) 원활해 진다. 이 기술이 갖추어져 있는 체계는 개별적인 화소위치를 화소직경의 소수부에 의하여 옮겨 지도록 설계한다. 전자속은 선의 실제경로 또는 물체경계에 더 가까운 점이 현시되도록 하기 위하여 일반적으로 화소직경의 1/4, 1/2, 3/4만큼 옮겨 진다. 일부 체계들은 또한 세기를 분포시키는 추가적인 방법으로서 개별적인 화소의 크기를 조정할수 있게 한다. 그림 4-41에 여러가지 선경로에서 화소위치조정에 의한 경계허상제거효과를 보여 주었다.

직선의 세기차에 대한 보상

선의 계단모양을 유연하게 하는 경계허상제거는 그림 4-42에서 보여 주는 다른 라스터영향들도 보상한다. 비록 선들이 다 같은 개수의 화소들로 현시된다하여도 대각선은 수평선보다 $\sqrt{2}$배 길다. 이것의 시각적인 효과는 수평선보다 약한 밝기로 나타난다. 왜냐하면 대각선은 단위길이당 더 낮은 세기로 현시되기때문이다. 선긋기알고리듬을 매선의 경사도에 따라 그의 세기가 조정되도록 하면 이 영향을 보상하게 할수 있다. 수평 및 수직선은 제일 낮은 세기로 현시되게 하고 45°선에는 제일 높은 세기가 주어 지도록 한다. 그러나 경계허상제거기술을 현시에 적용하면 세기들은 자동적으로 보상된다. 유한너비선을 고려함으로써 선들이 자기의 길이에 비례하는 총세기를 가지고 현시되도록 화소세기가 조정된다.

ㄱ) ㄴ)

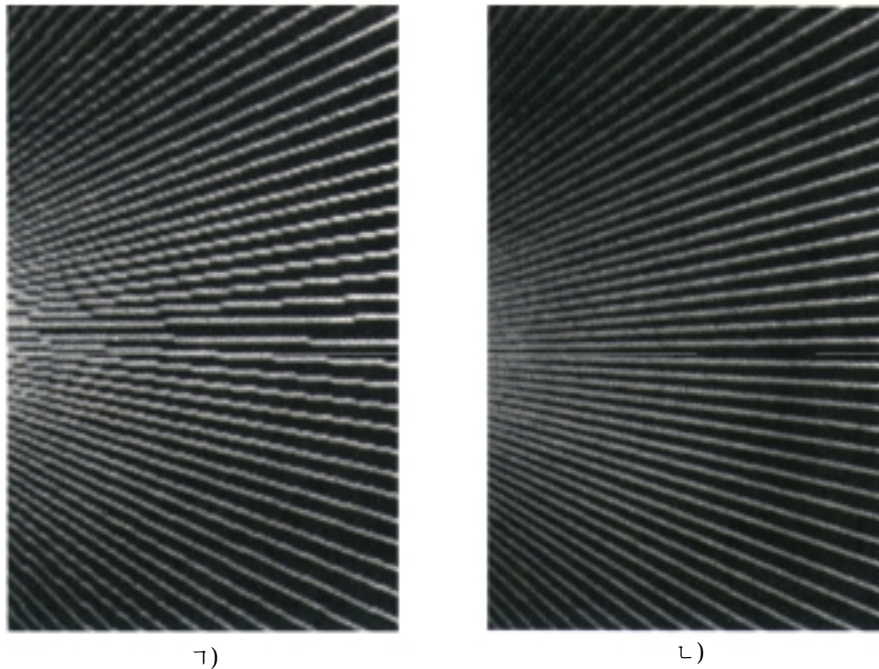



그림 4-41. Merlin 9200 체계에서 현시된 터실터실한 선(ㄱ)들은 화소위치조종이라고 하는 경계허상제거방법에 의하여 원활해 진다(ㄴ). 이 방법은 체계에서 주소화 가능한 점의 개수를 768 × 576 으로부터 3072 × 2304 로 증가시킨다.

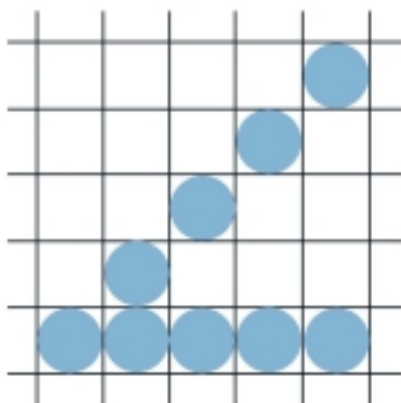



그림 4-42. 동일한 수의 화소로 현시되지만 길이가 같지 않은 두 선

구역경계의 경계허상제거

선에 대한 경계허상제거개념은 구역경계가 터실터실하게 나타나는것을 제거하는데도 적용할수 있다. 이 절차는 구역을 발생시킬 때에 구역의 륜곽선을 원활하게 하도록 주사선알고리듬에 합쳐 넣을수 있다.

체계가 화소의 재위치설정을 할수 있는 능력을 가지고 있다면 경계화소위치들을 구역의 경계를 정의하는 선을 따르도록 조정함으로써 구역의 경계를 원활하게 할수 있다. 다른 방법은 매개 화소의 세기를 경계위치에서는 경계안에 들어 가는 화소구역의 퍼센트에 따라 조정하는것이다. 그림 4-43에서 위치 (x, y)의 화소는 다각형경계안에서 자기의 약 절반구역을 차지한다. 따라서 그 위치에서의 세기는 거기에 할당되는 값의 절반으로 조정할수 있다. 경계를 따라 다음 위치 $(x+1, y+1)$에서 세기는 그 점에 할당되는 값의 약 1/3로 조정된다. 이런 방법은 화소구역적용범위의 퍼센트에 기초하여 경계주위의 다른 세기값들에도 적용된다.

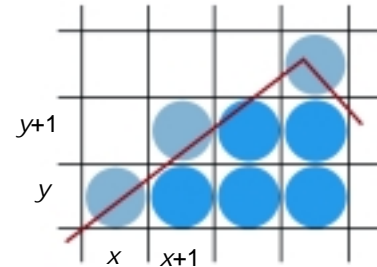



그림 4-43. 구역경계를 따르는 화소세기의 조정

초표본화방법은 전체 구역을 부분분할하고 구역경계안에 들어 가는 부분화소수를 결정하는 방법으로 적용할수 있다. 화소를 4개의 부분구역으로 갈라 놓은것을 그림 4-44에 보여 주었다. 본래의 4×4화소격자가 8×8격자로 되여 이제는 4개 대신에 이 격자를 지나는 8개 주사선을 처리한다. 그림 4-45에서는 이 격자에서 물체경계와 겹치는 하나의 화소구역을 보여 주고 있다. 두개의 주사선우에서 세개의 부분화소구역이 물체의 경계안에 있다는것을 알수 있다. 때문에 화소의 세기를 최대값의 75%로 설정한다.

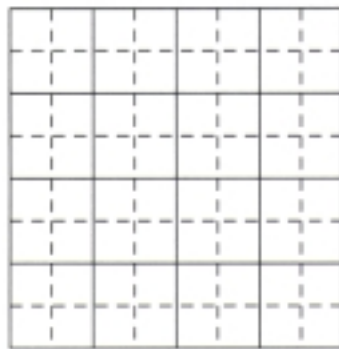



그림 4-44. 8×8 살창으로 부분분할된 라스터현시장치의 4×4 화소부분

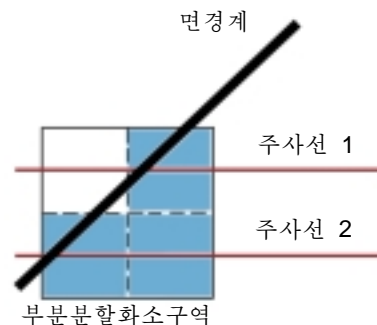



그림 4-45. 물체경계선안에 세개의 부분분할구역이 들어 가는 화소구역

Pitteway와 Watkinson에 의하여 개발된 화소구역의 경계안에서의 퍼센트를 결정하는 다른 방법은 선의 중점알고리듬에 기초하고 있다. 이 알고리듬은 선을 따라 다음화소를 두 후보화소들사이의 중간위치를 검사하고 두 화소중 어느것이 선에 더 가까운가를 결정하여 선택한다. 이때 브리센함의 알고리듬에서와 같이 부호가 두개의 후보화소중 어느것이 선에 더 가까운가를 말해 주는 결심파라메터 p를 설정한다. p의 형식을 약간 수정하면 또한 물체에 의해 덮여 지는 현재 화소구역의 퍼센트값을 주는 량을 얻게 된다.

먼저 경사도 m이 0∼1사이의 범위에 있는 선에 대하여 이 방법을 고찰하기로 하자. 그림 4-46에 화소격자에서의 직선경로를 보여 준다. (x_k, y_k)위치의 화소가 현시되였다고 하면 선에 제일 가까운

다음화소는. $x = x_k + 1$에서, y_k 또는 $y_k + 1$의 화소이다. 어느 화소가 더 가까운가는

$$y - y_{mid} = [m(x_k + 1) + b] - (y_k + 0.5) \qquad (4-7)$$

을 계산하여 결정할수 있다. 이 값은 선우의 실제 y자리표로부터 y_k와 $y_k + 1$의 화소사이의 중간점까지의 수직거리이다. 이 차계산값이 부이면 y_k의 화소가 선에 더 가깝다. 값이 정이면 $y_k + 1$의 화소가 더 가깝다. 이 계산은 $1-m$을 더함으로써 0∼1사이의 정의 수로 되도록 조정할수 있다.

$$P = [m(x_k + 1) + b] - (y_k + 0.5) + (1 - m) \qquad (4-8)$$

이제 $p < 1 - m$ 이면 y_k의 화소가 더 가깝고 $p > 1 - m$ 이면 $y_k + 1$ 의 화소가 더 가깝다.

파라메터 p는 또한 현재 화소에서 구역에 의하여 겹쳐 지는 량도 나타낸다. 그림 4-47 에서 (x_k, , y_k)의 화소에서 화소의 안쪽 부분은

$$\text{area} = mx_k + b - y_k + 0.5 \qquad (4-9)$$

와 같이 계산할수 있다. (x_k , y_k)에서의 화소겹침구역에 대한 이 식은 식 4-8의 결심파라메터 p에 대한것과 같다. 따라서 다각형경계를 따라 다음화소위치를 결정하기 위하여 p를 평가하는것은 또한 현재화소에 대한 구역적용범위의 퍼센트를 결정하는것으로도 된다.

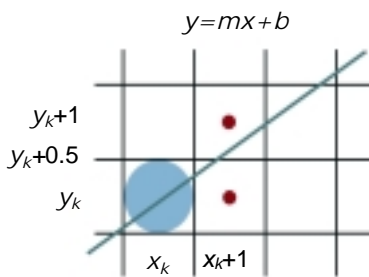



그림 4-46. 화소살창부분을 통과하는 구역의 경계선

이 알고리듬은 부의 경사도와 1보다 큰 경사도를 가지는 선들도 처리하게 일반화할수 있다. 파라메터 p에 대한 이 계산은 화소위치들과 물체경계위치를 지적하는것과 동시에 경계선을 따라 화소세기를 조정하도록 중점선알고리듬에 병합시킬수 있다. 또한 화소의 자리표를 그의 왼쪽 아래자리표로 참조되도록 계산을 조정하여 3장 10절에서 설명된바와 같이 구역의 비례를 유지하게 할수 있다.

그림 4-48에 보여 준바와 같이 다각형정점에서와 대단히 여윈 다각형에서는 하나의 화소구역을 통과하는 하나이상의 경계변을 가지게 된다. 이런 경우에는 하나의 화소를 통과하는 모든 변들을 처리하고 정확한 내부구역을 결정하도록 Pitteway-Watkinson알고리듬을 수정해야 한다.

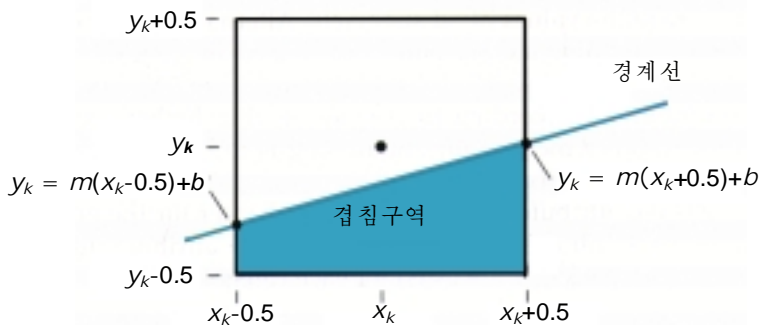



그림 4-47. 중심이 (x_k, y_k)에 있고 다각형구역이 포함되는 화소직 4 각형의 겹침구역

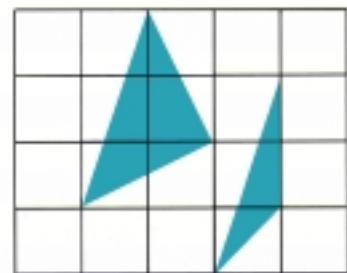



그림 4-48. 개별적인 화소구역을 하나 이상의 경계선이 통과하는 다각형

선의 경계허상제거에서 설명한 려파기술은 구역경계에도 적용할수 있다. 또한 여러가지 경계허상제거방법들도 다각형구역이나 곡선경계를 가지는 구역들에 적용할수 있다. 경계방정식은 화소구역과 현시될 구역의 구역겹침을 평가하는데 리용한다. 그리고 주사선과 주사선사이에서 상관성기술을 리용하면 계산이 간단해 진다.

요약

이 장에서는 출력되는 기초요소들의 외형을 조종하는 여러가지 속성들을 조사하였다. 기초요소들을 현시하는 절차들에서는 속성을 설정하여 직선발생, 구역채우기, 문자렬현시를 위한 알고리듬들의 출력을 조정한다.

선의 기본속성들로서는 선의 형태, 색, 너비이다. 선의 형태지적에는 실선, 파선, 점선이 속한다. 선의 색은 RGB현시장치에서 3개의 전자총의 세기를 조종하는 RGB구성성분들로 지적할수 있다. 선의 너비는 한 화소너비의 표준선에 대한 배수로 지적된다. 이런 속성들은 직선과 곡선들에 다같이 적용된다.

프레임완충기의 크기를 줄이기 위하여 일부 라스터체계들에서는 개별적인 색검색표를 사용한다. 이것은 색의 수를 제한시킨다. 즉 색을 검색표의 길이만큼 현시할수 있다. 완전색체계는 개별적인 색검색표를 쓰지 않고 화소당 24bit 제공한다.

채운구역속성에는 채우기류형, 채우기색 또는 채우기무늬가 들어 간다. 채우기류형이 옹근색으로 되는 경우 채우기색은 다각형내부의 옹근채우기에 대한 색갈을 가리킨다. 속이 빈 채우기류형인 경우는 내부를 배경색으로 하고 경계를 채우기색으로 한다. 세번째 채우기류형은 무늬채우기이다. 이 경우에는 선택된 배렬무늬가 다각형내부를 채우는데 쓰인다.

일부 프로그람들에서는 추가적인 채우기선택항목으로서 색배합채우기를 제공하고 있다. 이 채우기는 경계허상을 제거하는데서와 그림그리기프로그람들에서 리용된다. 색배합채우기절차들은 구역에 대한 새로운 채우기색을 이전의 채우기색과 같은 변화를 가지게 한다.이런 방법들의 한가지 실례는 선형적인 색배합채우기알고리듬인데 여기서는 이전의 채우기가 전경색과 배경색의 선형적인 결합으로 진행되였다고 가정한다. 이와 같은 선형적인 관계식을 프레임완충기로부터 결정하여 구역을 새로운 색으로 칠하는데 리용한다.

화소격자무늬나 륜곽선폰트로 정의되는 문자들은 각이한 색갈, 크기, 방향으로 현시할수 있다. 문자렬의 방향을 설정하기 위하여 문자웃벡토르방향과 본문경로방향을 설정한다. 보충적으로 문자렬의 기준맞추기를 시작자리표위치에 대하여 설정할수 있다. 표식기호는 여러가지 크기와 색갈의 문자를 선택하여 현시할수 있다.

도형처리프로그람들에서는 묶은것과 묶지 않은 속성을 다같이 지적할수 있게 되여 있을수 있다. 묶지 않은 속성들은 오직 하나의 출구장치에 대하여 정의하는것들이다. 묶음속성지적은 각이한 장치들에서 같은 묶음표첨수번호를 가지고 호출되는 각이한 속성들의 묶음을 쓸수 있게 한다. 이미 갖추어 져 있을수도 있고 사용자가 정의할수도 있으며 둘다 있을수도 있다. 묶음표값을 설정하기 위한 함수들은 워크스테이션의 형과 주어 진 속성첨수에 대한 속성목록을 지적한다.

속성과 기타 파라메터들에 대한 현재설정을 알아 보기 위하여서는 질문함수를 리용할수 있다. 질문함수로는 색과 기타 속성정보를 꺼내 보는것과 함께 워크스테이션코드, 상태값들을 얻을수 있다.

주사변환은 라스터체계에서 수자화하는 과정이다. 때문에 현시되는 기초요소들은 터실터실한 걸모양을 가진다. 이것은 자리표값들을 화소위치값으로 둥그리기하는 정보의 거친표본화때문이다. 화

소세기들을 조정하는 경계허상제거절차들을 적용함으로써 라스터기초요소들의 걸모양을 개선할수 있다. 이것을 수행하는 한가지 방법은 초표본화하는것이다. 즉 매개 화소를 부분화소들로 이루어 진것으로 보고 부분화소들의 세기를 계산하고 모든 부분화소값들을 평균한다. 다른 한가지는 구역표본화를 진행하고 화면의 화소들에 대하여 구역의 덮음범위의 퍼센트를 결정한 다음 이 퍼센트에 비례하는 세기를 화소에 설정한다. 또한 중심부분화소들에 더 많은 무게를 주도록 부분화소들의 위치에 따라 기여하는 몫에 무게를 줄수 있다. 경계허상을 제거하는 다른 방법은 화소위치들을 밀어 놓을수 있게 장치구성을 특별하게 하는것이다.

표 4-4에 이 장에서 설명한 출력기초요소들의 속성들을 직선, 채운구역, 본문, 표식 등 도형처리 프로그람들에서 리용할수 있는 속성함수들을 부류별로 적어 놓았다.

표 4-4. 속성들에 대한 종합

요소형	관련속성	속성설정함수	묶음속성함수
Line	Type	Setlinetype	SetpolylineIndex
	Width	SetlinewidthScaleFactor	SetpolylineRepresentatoin
	Color	SetpolylineColourIndex	
Fill Area	Fill Style	SetInteriorStyle	SetInteriorIndex
	Fill Color	SetInteriorColorIndex	SetInteriorRepresntation
	Pattern	SetInteriorStyleIndex	
		SetPatternRepresentation	
		SetPatternSize	
		SetPatternReferencePoint	
Text	Font	SetTextFont	SetTextIndex
	Color	SetTextColourIndex	SetTextRepresentation
	Size	SetCharacterHeight	
		SetCharacterExpansionFactor	
	Orientation	SetCharacterUpVector	
		SetTextPath	
		SetTextAlignment	
Marker	Type	SetMarKerType	SetPolymarKerIndex
	Size	SetMarKerSizeScaleFactor	SetPolymarKerRepresentation
	Color	SetPolymarKerColourIndex	

참고문헌

색 및 회색계조에 대한 고찰은 Crow(1978)와 Heckbert(1982)에서 설명하였다. 색배합채우기기술은 Fishkin과 Barsky(1984)에서 주고 있다.

경계허상제거기술은 Pitteway 과 Watkinson(1980), Crow(1981), Turkowski (1982), Korein 과 Badle (1983), Kirk 와 Avro, Schilling, Wu(1991)에서 설명하였다.

PHIGS에서의 속성함수들은 Howard(1991)들과 Hopgood와 Duce(1991), Gaskins(1992), Blake(1993)에서 설명하였다. GKS워크스테이션과 속성정보에 대하여서는 Hopgood(1983)들과 Enderle, Kansy, Pfaff(1984)를 보시오.

련습문제

4-1. 브리센함선긋기알고리듬을 수정하여 실선, 파선, 점선을 현시하는 선형태함수를 실현하시오.

4-2. 선의 중점알고리듬으로 실선, 파선, 점선을 어느것이나 현시할수 있게 선형태함수를 실현하시오.

4-3. 선형태함수를 실현하기 위한 병렬적인 방법을 제기하시오.

4-4. 선두께함수를 실현하기 위한 병렬적인 방법을 제기하시오.

4-5. 두개의 끌점과 두께로 지적된 선은 4개의 정점을 가진 직4각형으로 변환될수 있으며 그렇게 되면 주사선법을 써서 현시할수 있다. 선의 끌점들과 너비값을 리용하여 직4각형을 정의하는데 필요한 4개 정점을 계산하기 위한 효과적인 알고리듬을 전개하시오.

4-6. 선긋기프로그람에서 선너비함수를 실현하며 세가지 선너비를 현시할수 있게 하시오.

4-7. 동일한 x자리표구간에서 정의된 세가지 자료모임의 선그라프를 출력하기 위한 프로그람을 쓰시오. 프로그람에 대한 입력은 3개 조의 자료값들과 자리표축들의 표식 그리고 화면에서 현시될 구역에 대한 자리표들이다.

4-8. 절단모자, 둥근모자, 투영4각형모자를 가지고 두꺼운 선을 현시하기 위한 알고리듬을 작성하시오. 이 선택항목들은 선택항목차림표에서 제공되게 된다.

4-9. 연귀련결, 둥근련결, 경사련결로 두꺼운 절선을 현시하기 위한 알고리듬을 제기하시오. 이 선택항목들은 선택항목차림표에서 제공되게 된다.

4-10. 선그리기절차에 대한 펜 및 붓을 적어도 2개의 선택항목 즉 원형과 정4각형으로 가지는 차림표선택항목을 실현하시오.

4-11. 출력되는 선의 세기가 그것들의 경사에 의하여 설정되게끔 선긋기알고리듬을 수정하시오. 즉 경사도값에 따라 화소세기를 조정함으로써 모든 선들이 단위길이당 같은 세기를 가지고 현시되도록 하시오.

4-12. 현시되는 타원의 선형태(실선, 파선, 점선)를 조종하기 위한 함수를 정의하고 실현하시오.

4-13. 현시되는 타원의 너비를 설정하기 위한 함수를 정의하고 실현하시오.

4-14. 임의로 지적되는 화면구역에 기둥도표를 현시하기 위한 루틴을 쓰시오. 포함시켜야 할 입력으로서는 자료모임, 자리표축들에 대한 표식, 화면구역에 대한 자리표이다.

4-15. 동일한 x자리표구간에서 정의되는 2개의 자료모임을 지적된 현시화면구역에 꼭 맞도록 비례변환하여 현시하기 위한 프로그람을 쓰시오.

4-16. 색검색표와 setColourRepresentation연산을 실현하는 알고리듬을 만드시오.

4-17. 인치당 100화소의 8in×10in영상화면을 가진 체계가 있다. 만약 64개 위치를 가진 색검색표가 이 체계에서 사용되고 있다면 프레임완충기의 크기(byte로)가 최소한 얼마로 되겠는가?

4-18. 화소당 20bit 의 512×512 의 프레임완충기와 화소당 24 bit 의 색검색표를 가진 RGB 라스

터체계를 생각하자.

ㄱ) 이 체계에서 서로 다른 회색계조를 얼마나 현시할수 있는가?

ㄴ) 회색계조를 포함하여 얼마만한 색들을 현시할수 있는가?

ㄷ) 한번에 얼마만한 색을 현시할수 있는가?

ㄹ) 기억기의 총체적인 크기는 얼마인가?

ㅁ) 색능력이 같으면서도 기억기를 줄일수 있는 두가지 방법을 설명하시오.

4-19. 임의의 지적된 직4각형채우기무늬로 설계되는 무늬의 위치로부터 시작하여 다각형내부에 적용할수 있게 주사선알고리듬을 수정하시오.

4-20. 지적된 무늬로 주어 진 타원의 내부를 채우기 위한 프로그람을 쓰시오.

4-21. setPatternRepresentation 함수를 실현하기 위한 프로그람을 쓰시오.

4-22. 이미 있는 직4각형채우기무늬의 크기를 변화시키기 위한 절차를 정의하고 실현하시오.

4-23. 색배합채우기알고리듬을 실현하는 프로그람을 쓰시오. 무엇이 색배합채우기알고리듬을 완성시키며 어떤 색들이 결합되는가를 신중히 규정하시오.

4-24. 직4각형격자무늬로 정의되는 문자의 높이와 너비를 조정하기 위한 알고리듬을 제기하시오.

4-25. 문자렬현시를 조종하는 문자의 웃벡토르와 본문경로를 설정하기 위한 루틴을 실현하시오.

4-26. 기준맞추기파라메터에 대한 입력값을 지적하여 본문을 정렬시키는 프로그람을 쓰시오.

4-27. 표식속성함수를 실현하는 프로그람을 전개하시오.

4-28. 묶음속성을 쓰는 체계에서 요구되는 속성실현절차와 묶지 않은 속성을 쓰는 체계에 요구되는 속성실현절차를 비교하시오.

4-29. 묶지 않은 체계의 속성표에 속성들을 기억시키고 호출하기 위한 프로그람을 전개하시오. 프로그람들은 속성들을 해당한 출력루틴들에 넘겨 주며 질문명령에서 지적되는 기억기위치에 속성들을 넘겨 주기 위하여 설계되는 속성값들이 체계표에 기억되게끔 설계된다.

4-30. 앞의 련습문제에서 서술된것과 같은 프로그람을 묶음속성표체계에 대하여 만드시오.

4-31. 직선경로근방에서의 화소세기들을 조정할수 있게 브리센함직선알고리듬을 확장하여 경계허상을 제거하는 절차를 실현하시오.

4-32. 선의 중점알고리듬으로 경계허상을 제거하는 프로그람을 쓰시오.

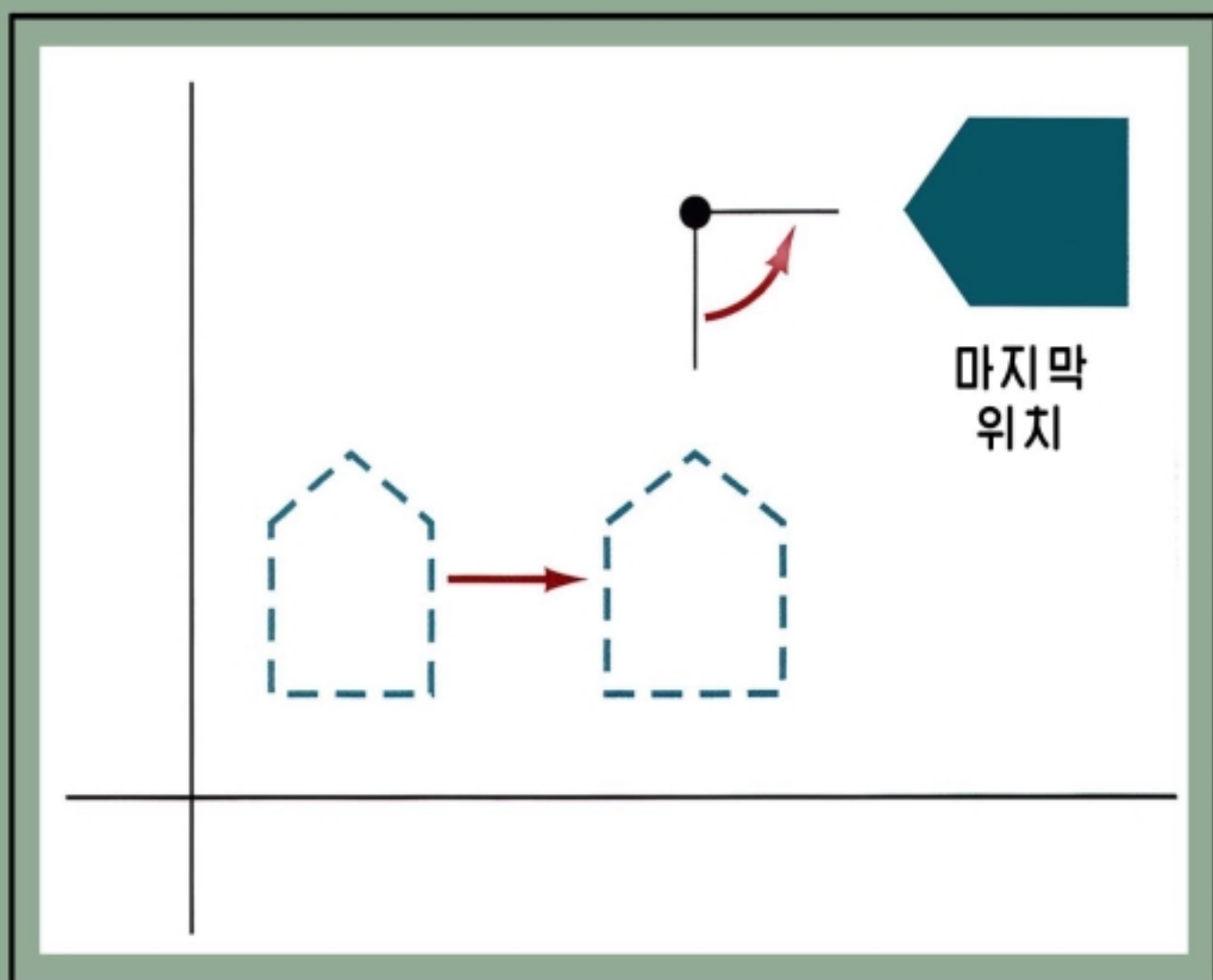
4-33. 타원경계의 경계허상을 제거하는 알고리듬을 전개하시오.

4-34. 구역채우기에 대한 주사선알고리듬을 경계허상도 제거할수 있게 수정하시오. 린접한 주사선에서의 계산을 줄이기 위하여 상관성수법을 쓰시오.

4-35. Pitteway-Watkinson경계허상제거알고리듬을 다각형내부를 채우기 위한 주사선절차로 실현하는 프로그람을 쓰시오. 프레임완충기의 (x, y)위치에 세기값을 넣는데

setPixel(x, y, intensity)

루틴을 쓰시오.

5장. 2차원기하학적변환

여러가지 그림과 그라프는 출력기초요소들과 그 속성들을 현시하는 절차들에 의하여 만들어 낼 수 있다. 그런데 대부분의 응용분야들에서는 화면의 현시를 바꾸거나 화면에 대하여 처리를 진행하는 경우도 있다. 설계도와 간단한 배치도는 장면의 요소부분들의 방향과 크기를 맞추어 만든다. 그리고 동화는 카메라나 장면안의 물체를 동화경로를 따라 움직여서 만든다. 방향, 크기, 형태의 이런 변화는 물체의 자리표서술을 변화시키는 기하학적인 변환에 의하여 수행된다. 기본적인 기하학적인 변환은 평행이동, 회전, 비례변환이다. 물체에 자주 적용되는 기타 변환들로서는 반사와 쏠림이 있다. 여기서는 먼저 기하학적인 변환을 수행하는 방법들을 설명하고 변환함수들을 도형처리프로그람에 어떻게 병합시킬수 있겠는가를 고찰한다.

1절. 기본변환

먼저 평행이동, 회전, 비례변환과 같은 파라메터들을 써서 2차원물체의 위치와 크기를 다시 정하는 일반적인 절차를 설명한다. 다음에 2절에서 물체의 변환들을 효과적으로 결합할수 있도록 변환식들을 보다 편리한 행렬형식으로 어떻게 표현할수 있는가를 고찰한다.

평행이동

평행이동(translation)은 물체를 직선경로를 따라 한 자리표위치에서 다른데로 다시 위치정하게 한다. 2차원적인 점을 새로운 위치 (x', y')에로 움직이기 위하여서는 본래 자리표위치 (x, y)에 평행이동거리 t_x, t_y를 더하면 된다(그림 5-1).

$$x' = x + t_x, \qquad y' = y + t_y \tag{5-1}$$

평행이동거리쌍 (t_x, t_y)를 **평행이동벡토르** 또는 **밀기벡토르**(shift vector)라고 한다.

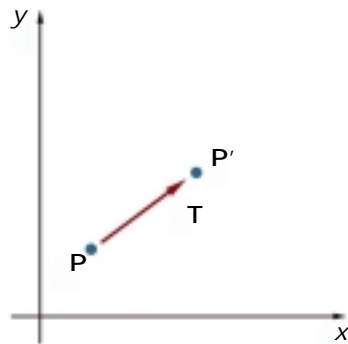



그림 5-1. 평행이동벡토르 **T**에 의하여 점을 위치 **P**로부터 **P′**로 평행이동시킨다.

식 5-1은 자리표위치와 평행이동벡토르를 표현하는 렬벡토르를 리용하여 하나의 행렬방정식으로 표현할수 있다.

$$\mathbf{P} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad \mathbf{P}' = \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{5-2}$$

이를 리용하여 2차원평행이동식을 행렬형식으로 쓰면

$$\mathbf{P}' = \mathbf{P} + \mathbf{T} \tag{5-3}$$

때때로 행렬변환식을 자리표의 렬벡토르대신에 행벡토르에 의해서도 표현한다. 이 경우에 행렬표현을 P=$[x, y]$, T=$[t_x, t_y]$로 쓸수 있다. 점에 대한 렬벡토르표현은 표준수학표기이고 많은 도형처리프로그람들 실례로 GKS와 PHIGS에서도 렬벡토르표현을 리용하기때문에 여기에서도 그렇게 하기로 한다.

평행이동은 물체를 변형시킴이 없이 움직이는 강체변환이다. 즉 물체의 모든 점이 동일한 크기만큼 평행이동된다. 선분은 선의 매 끝점들에 변환식 5-3을 적용하고 새 끝점위치들사이에 선을 다시 긋는것에 의하여 평행이동된다. 다각형은 평행이동벡토르를 매 정점의 자리표위치에 더하고 정점자리표들의 새 모임과 현재 속성설정을 리용하여 다각형을 다시 발생시키면 평행이동된다. 그림 5-2는 물체를 한 위치에서 다른 위치로 움직이는 지적된 평행이동벡토르의 응용을 설명한다.

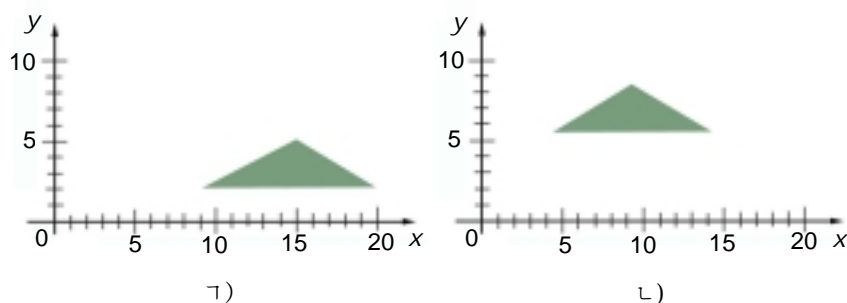



그림 5-2. 다각형을 ㄱ위치로부터 ㄴ위치에로 평행이동벡토르 (-5.50, 3.75)에 의하여 평행이동시킨다.

이런 방법은 곡선물체를 평행이동시키는데도 리용된다. 원 또는 타원의 위치를 변화시키기 위하여 중심자리표를 평행이동시키고 새 위치에서 그림을 다시 그린다. 다른 곡선(실례로 스플라인)들은 물체를 정의하는 자리표위치들을 평행이동시키고 다음에 평행이동된 자리표점들을 리용하여 곡선경로를 복구한다.

회전

2 차원회전은 물체를 xy평면에서 원경로를 따라 위치를 다시 정하게 한다. 회전을 발생시키기 위하여서는 **회전각** θ와 물체가 회전하게 될 **회전중심점**(또는 **회전축점**)의 위치 (x_r, y_r)를 지적한다(그림 5-3). 그림 5-3 에서와 같이 회전각이 정의 값을 가지면 회전중심점에 대하여 시계바늘반대방향으로 회전되고 부의 값이면 물체가 시계바늘방향으로 회전된다. 이 변환은 또한 xy평면에 수직인 회전축점을 통과하는 회전축에 대한 회전이라고 표현할수 있다.

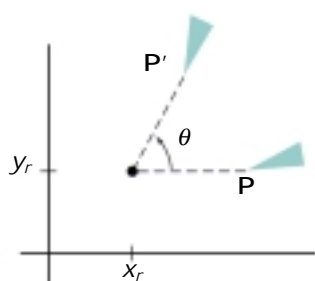



그림 5-3. 물체를 회전축점 (x_r, y_r)에 대하여 각 θ만큼 회전시킨다.

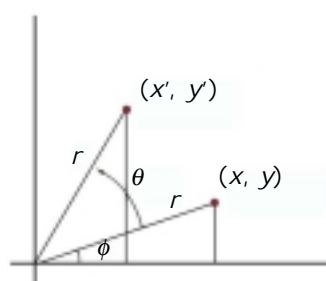



그림 5-4. 점을 자리표원점에 대하여 위치 (x, y)로부터 위치 (x', y')에로 각 θ만큼 회전시킨다. x축으로부터의 점의 초기방향각은 ϕ이다.

먼저 회전축점이 자리표원점에 있을 때 점 P의 회전에 대한 변환식을 결정하자. 점의 처음과 변환된 위치들의 방향각과 자리표들을 그림 5-4 에 보여 주었다. 이 그림에서 r는 원점으로부터 점까지의 거리상수이고 각 ϕ는 수평선으로부터의 점의 초기방향각, θ는 회전각이다. 표준삼각항등식을 리용하면 변환된 자리표들을 각 θ와 ϕ에 의하여

$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta$$
$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta \quad (5-4)$$

와 같이 표현할수 있다. 점의 본래 자리표는 극자리표로

$$x = r\cos\phi, \quad y = r\sin\phi \quad (5-5)$$

이다. 식 5-5를 5-4에 대입하면 위치 (x, y)의 점을 원점에 대하여 각 θ만큼 회전시키는 변환방정식을 얻는다.

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta \quad (5-6)$$

자리표위치에 대한 렬벡토르표현 5-2에 의하여 회전방정식을 행렬형식으로 다음과 같이 쓸수 있다.

$$\mathbf{P}' = \mathbf{R}\cdot\mathbf{P} \quad (5-7)$$

여기서

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \quad (5-8)$$

는 회전행렬이다.

자리표위치를 렬벡토르대신에 행벡토르로 표현하면 회전방정식 5-7의 행렬곱하기는 변환된 행자리표벡토르 $[x', y']$가

$$\mathbf{P}'^T = (\mathbf{R}\cdot\mathbf{P})^T$$
$$= \mathbf{P}^T\cdot\mathbf{R}^T$$

와 같이 계산되도록 전위된다. 여기서 $\mathrm{P}^T = [x \quad y]$이고 행렬 R 의 전위 R^T는 행과 렬을 서로 교체하여 얻는다. 회전행렬에 대한 전위는 시누스항들의 부호만 간단히 바꾸면 얻어 진다.

임의의 위치의 회전축에 대한 점의 회전은 그림 5-5에서 설명된다. 이 그림의 삼각관계를 리용하면 임의로 지적된 회전축위치 (x_r, y_r)에서의 점의 회전에 대한 변환방정식을 식 5-6 을 일반화하여 다음과 같이 얻을수 있다.

$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$
$$y' = y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta \quad (5-9)$$

이 일반화된 회전방정식은 자리표값들에서의 곱하기뿐아니라 더하기도 가지고 있어 식 5-6과 다르다. 그러므로 행렬식 5-7에 성분들이 식 5-9의 더하기(평행이동)항과 같은 렬벡토르를 더하여 회전축자리표가 포함되도록 수정할수 있다. 행렬방정식을 형식화하는 더 좋은 방법에 대하여서는 5장 2절에서 설명한다.

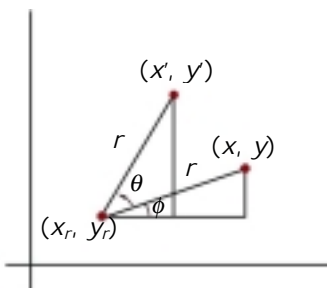



그림 5-5. 회전중심점 (x_r, y_r)에 대하여 점을 (x, y)위치로부터 (x', y')위치로 θ만한 각으로 회전시키는 경우

평행이동과 마찬가지로 회전은 물체를 변형시킴이 없이 움직이는 강체변환이다. 선분의 회전은 선의 매 끌점들에 회전방정식 5-9를 적용하고 새 끌점위치들사이에 선을 다시 그으면 된다. 다각형의 회전은 매 정점을 지적된 회전각만큼 옮겨놓고 새 정점들을 리용하여 다각형을 다시 그리면 된다. 곡선은 그를 정의하는 점들을 다시 위치정하고 곡선을 다시 그리는것에 의하여 회전된다. 실례로 원 또는 타원은 중심위치를 지적된 회전각에 대한 호를 따라 움직이는것에 의해 비중심축에 대하여 회전시킬수 있다. 타원인 경우에는 중심자리표에 대하여 주축과 부축을 회전시키는것에 의해 회전시킬수 있다.

비례변환

비례변환(scaling)은 물체의 크기를 변화시킨다. 이 조작은 다각형의 매 정점의 자리표값 (x, y)에 비례결수 s_x와 s_y를 곱하여 변환된 자리표 (x', y')를 얻는다.

$$x' = x \cdot s_x, \qquad y' = y \cdot s_y \qquad (5-10)$$

비례결수 s_x는 물체를 x방향으로 비례변환하고 s_y는 y방향으로 비례변환한다. 변환방정식 5-10은 또한 행렬형식으로 다음과 같이 쓸수 있다.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \qquad (5-11)$$

또는

$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P} \qquad (5-12)$$

여기서 **S**는 식 5-11의 2×2비례변환행렬이다.

비례결수 s_x 및 s_y에는 임의의 정의 수값이 할당될수 있다. 1보다 작은 값은 물체의 크기를 줄이고 1보다 큰 값은 확장시킨다. s_x와 s_y에 대하여 다같이 값 1을 지적하면 물체의 크기를 변화시키지 않는다. s_x와 s_y에 같은 값이 할당되는 경우는 물체의 상대적인 비례관계가 유지되게 된다. s_x 및 s_y의 값이 같지 않으면 비례변환 및 위치선정변환에 의하여 조정될수 있는 기초적인 형태들이 만들어 지는데 설계에서 자주 리용된다(그림 5-6).

ㄱ)

ㄴ)

그림 5-6. 바른 4각형(ㄱ)을 비례결수 s_x=2, s_y=1에 의해 직4각형(ㄴ)으로 변환시킨다.

식 5-11에 의해 변환되는 물체들은 크기도 변하고 위치도 다시 정해 진다. 1보다 작은 값을 가지는 비례결수는 물체를 자리표원점에 더 가깝게 움직이게 하며 1보다 큰 값은 자리표위치를 원점으로부터 더 멀리 움직이게 한다. 그림 5-7은 식 5-11에서 s_x와 s_y에 다같이 값 0.5를 주었을 때의 선의 비례변환을 설명한다. 선의 길이와 원점으로부터의 거리는 다같이 비례결수 1/2에 의하여 작아 진다.

그림 5-7. 식 5-12에서 $s_x = s_y = 0.5$를 리용하여 비례변환된 선은 치수가 작아 지고 원점에 더 가깝게 움직인다.

비례변환되는 물체의 위치는 비례변환후에 변화되지 않고 남아 있는 **고정점**이라고 부르는 위치를 선택하는 방법으로 조종할수 있다. 고정점 (x_f, y_f)의 자리표는 정점, 물체의 중심, 임의의 다른 위치중 하나로 선택될수 있다(그림 5-8). 다각형인 경우는 고정점에 대하여 매 정점으로부터 고정점까지의 거리를 비례변환하는것에 의해 비례변환다각형을 얻는다. 자리표가 (x, y)인 정점에 대하여 비례변환된 자리표 (x', y')는

$$x' = x_f + (x - x_f)s_x, \quad y' = y_f + (y - y_f)s_y \qquad (5-13)$$

그림 5-8. 선택된 고정점 (x_f, y_f)에 대한 비례변환(다각형의 매 정점으로부터 고정점까지의 거리가 변환식에 의하여 비례변환된다.)

와 같이 계산된다. 이 비례변환은 곱하기와 더하기항을 분리시켜 다음과 같이 다시 쓸수 있다.

$$x' = x \cdot s_x + x_f(1 - s_x)$$
$$y' = y \cdot s_y + y_f(1 - s_y) \qquad (5-14)$$

여기서 더하기항 $x_f(1-s_x)$와 $y_f(1-s_y)$ 는 물체의 모든 점에 대하여 상수이다.

비례변환식에 고정점에 대한 자리표가 포함된것은 회전변환식에 회전축점에 대한 자리표가 포함된것과 류사하다. 구성성분들이 식 5-14의 상수항들인 렬벡토르를 설정하고 이 렬벡토르를 식 5-12의 적 S·P에 더한다. 다음 부분에서 변환방정식에 행렬곱하기만 포함되는 행렬형식화에 대하여 설명한다.

다각형은 매 정점에 식 5-14의 변환을 적용하고 변환된 정점들을 리용하여 다각형을 다시 발생시키면 비례변환된다. 다른 물체들은 물체를 정의하는 파라메터들에 비례변환식을 적용하는 방법으로 비례변환한다. 표준위치의 타원은 반주축 및 반부축을 비례변환하고 타원을 지적된 중심자리표에 대하여 다시 그림으로써 크기가 새롭게 설정된다. 원의 비례변환은 간단히 원의 반경을 조정한 다음 중심자리표에 대하여 변환된 반경을 리용하여 원을 다시 그린다.

2절. 행렬표현 및 동차자리표

많은 도형처리응용들에서는 일련의 기하학적인 변환들이 차례로 련속적으로 진행된다. 실례로 동화는 운동의 매 걸음에서 물체가 평행이동 및 회전될것을 요구한다. 설계와 그림그리기응용에서는 그림요소를 적당한 위치에 방향을 맞추어 놓기 위하여 평행이동, 회전, 비례변환을 진행한다. 이 절에서는 이러한 변환순서렬을 효과적으로 처리할수 있도록 앞에서 설명한 행렬표현들을 어떻게 재형식화할수 있는가를 고찰한다.

1 절에서는 매개 기본변환들이 일반행렬형식

$$\mathbf{P}' = \mathbf{M}_1 \cdot \mathbf{P} + \mathbf{M}_2 \qquad (5-15)$$

으로 표현될수 있다는것을 보았다. 여기서 자리표위치 **P**와 **P**′는 렬벡토르로 표현된다. 행렬 $\mathbf{M}_1$은 곱하기요소가 들어 있는 2×2배렬이며 $\mathbf{M}_2$는 이동항을 포함하는 두 원소행렬이다.

평행이동변환에서는 $\mathbf{M}_1$이 단위행렬이다. 회전과 비례변환에서는 $\mathbf{M}_2$에 회전축점 또는 비례변환고정점과 관련되는 평행이동항이 포함된다. 비례변환 그다음 회전 그리고 또 평행이동과 같은 변환순서렬을 만들기 위해서는 이 식에 의해 변환되는 자리표들을 한번에 한걸음씩 계산하여야 한다. 먼저 자리표위치들을 비례변환하고 다음에 이 비례변환된 자리표들을 회전시키고 마지막에 회전된 자리표들을 평행이동시킨다. 보다 효률적인 방법은 마지막자리표위치가 초기자리표로부터 직접 얻어 지도록 변환들을 결합하여 중간자리표값의 계산을 없애는것이다. 이렇게 하기 위해서는 $\mathbf{M}_2$의 평행이동항과 관련된 행렬더하기를 없애도록 식 5-15를 재형식화하여야 한다.

2차원기하학적변환에서 곱하기와 평행이동항은 2×2행렬표현을 3×3행렬로 확장함으로써 하나의 행렬표현으로 결합시킬수 있다. 이것은 모든 변환식들이 행렬곱하기로 표현되게 하며 또한 자리표위치에 대한 행렬표현을 확장시킨다. 2차원변환들을 행렬곱하기로 표현하기 위하여 매개 직각자리표위치 (x, y)를 동차자리표 (x_h, y_h, h)로 표현한다. 여기서

$$x = \frac{x_h}{h}, \qquad y = \frac{y_h}{h} \qquad (5-16)$$

그러므로 일반동차자리표표현은 또한 $h \cdot x$, $h \cdot y$, h와 같이 쓸수 있다. 2차원기하학적변환에서 동차파

라메터 h는 임의의 령아닌 값으로 선택할수 있다. 그러므로 매 자리표점 (x, y)에 대하여 무한히 많은 동등한 동차표현이 있게 된다. 간단히 h=1로 설정하면 편리하다. 그러면 매개 2차원위치는 동차자리표 $(x, y, 1)$로 표현된다. 파라메터 h에 대한 다른 값들도 필요하다. 실례로 3차원보기변환의 행렬형식화를 들수 있다.

동차자리표(homogeneous coordinates)라는 용어는 수학에서 이 표현의 직각자리표식들에서의 결과를 가리키는데 리용된다. 직각자리표 점 (x, y)가 동차표현 (x_h, y_h, h)로 변환될 때 $f(x, y)$와 같이 x와 y를 포함하는 식들은 이 3개의 파라메터 x_h, y_h, h의 동차식으로 된다. 이것은 바로 3개의 매 파라메터를 임의로 h배하여 바꾸어 넣으면 값 h를 식들의 밖으로 인수분해할수 있다는것을 의미한다.

위치를 동차자리표로 표현하면 모든 기하학적변환식들이 행렬곱하기로 표현되게 된다. 자리표는 3개 원소의 렬벡토르로 표현되며 변환연산은 3×3행렬로 씌여 진다. 평행이동에 대하여서는

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (5-17)$$

이것은 생략된 형식

$$\mathbf{P}' = \mathbf{T}(t_x, t_y) \cdot \mathbf{P} \qquad (5-18)$$

으로 쓸수 있다. 여기서 $\mathbf{T}(t_x, t_y)$는 식 5-17의 3×3평행이동행렬이다. 역평행이동행렬은 평행이동파라메터 t_x, t_y를 부로 즉 $-t_x$ 및 $-t_y$로 교체하면 얻어 진다.

류사하게 자리표원점에 대한 회전변환식은

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (5-19)$$

로 씌여 진다. 또는

$$\mathbf{P}' = \mathbf{R}(\theta) \cdot \mathbf{P} \qquad (5-20)$$

회전변환연산자 $\mathbf{R}(\theta)$는 회전파라메터 θ를 가지는 식 5-19의 3×3행렬이다. θ를 $-\theta$로 교체하면 역회전행렬이 얻어 진다.

마지막으로 자리표원점에 대한 비례변환은 행렬곱하기

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (5-21)$$

또는

$$\mathbf{P}' = \mathbf{S}(s_x, s_y) \cdot \mathbf{P} \qquad (5-22)$$

로 표현된다. 여기서 $\mathbf{S}(s_x, s_y)$는 파라메터 s_x와 s_y를 가지는 식 5-21의 3×3행렬이다. 이 파라메터들을 역수 $(1/s_x, 1/s_y)$로 교체하면 역비례변환행렬이 얻어 진다.

행렬표현은 도형처리체계들에서 변환을 실현하는 표준방법이다. 많은 체계들에서 회전 및 비례변환함수는 식 5-19와 5-21에서와 같이 자리표원점에 대한 변환이다. 다른 참조점에 대한 회전 및 비례변환은 변환연산의 련속으로 처리한다. 다른 방법은 프로그람들에서 변환함수들에 비례변환고정점자리표와 회전축점자리표에 대한 파라메터를 주는것이다. 그러면 회전축점 또는 고정점을 포함하는 일반회전 및 비례변환행렬은 변환함수의 련속실행이 필요없이 직접 설정된다.

3절. 합성변환

앞절의 행렬표현에 의하여 임의의 순서의 변환들에 대한 행렬은 개별적인 변환의 행렬적을 계산하여 합성변환행렬로 설정할수 있다. 변환행렬의 적을 형성하는것을 흔히 행렬의 결합 또는 합성이라고 한다.

합성변환은 자리표위치들이 렬행렬표현인 경우 행렬들을 오른쪽에서 왼쪽 순서로 곱하는 방법으로 얻는다. 즉 련속되는 매 변환행렬은 선행한 변환행렬들의 적을 왼쪽에서 곱한다.

평행이동

련속된 두개의 평행이동벡토르 (t_{x1}, t_{y1}) 및 (t_{x2}, t_{y2})이 자리표위치 P에 적용된다면 변환된 마지막 위치 $\mathbf{P}'$는

$$\begin{aligned}\mathbf{P}' &= T(t_{x2}, t_{y2}) \cdot \{\mathbf{T}(t_{x1}, t_{y1}) \cdot \mathbf{P}\} \\ &= \{\mathbf{T}(t_{x2}, t_{y2}) \cdot \mathbf{T}(t_{x1}, t_{y1})\} \cdot \mathbf{P}\end{aligned} \qquad (5-23)$$

로 계산된다. 여기서 $\mathbf{P}$와 $\mathbf{P}'$는 동차자리표렬벡토르로 표현된다. 이 결과는 두개의 련속한 묶기에 대한 행렬적을 계산하면 검증할수 있다. 또한 평행이동의 이 순서렬에 대한 합성변환행렬은

$$\begin{bmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x1}+t_{x2} \\ 0 & 1 & t_{y1}+t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-24)$$

또는

$$\mathbf{T}(t_{x2}, t_{y2}) \cdot \mathbf{T}(t_{x1}, t_{y1}) = \mathbf{T}(t_{x1}+t_{x2}, t_{y1}+t_{y2}) \qquad (5-25)$$

이다. 이것은 두개의 련속한 평행이동은 더하기라는것을 보여 준다.

회전

점 $\mathbf{P}$에 적용되는 두개의 련속한 회전은 변환된 위치

$$\begin{aligned}\mathbf{P}' &= \mathbf{R}(\theta_2) \cdot \{\mathbf{R}(\theta_1) \cdot \mathbf{P}\} \\ &= \{\mathbf{R}(\theta_2) \cdot \mathbf{R}(\theta_1)\} \cdot \mathbf{P}\end{aligned} \qquad (5-26)$$

를 만든다. 두개의 회전행렬을 곱해 보면 두개의 련속한 회전은 더하기라는것을 검증할수 있다.

$$\mathbf{R}(\theta_2) \cdot \mathbf{R}(\theta_1) = \mathbf{R}(\theta_1 + \theta_2) \qquad (5-27)$$

그러므로 마지막으로 회전된 자리표는 합성회전행렬에 의하여

$$\mathbf{P}' = \mathbf{R}(\theta_1 + \theta_2) \cdot \mathbf{P} \qquad (5-28)$$

와 같이 계산할수 있다.

비레변환

련속된 두개의 비레변환연산에 대한 변환행렬들을 련결시키면 다음의 합성비레변환행렬이 얻어진다.

$$\begin{bmatrix} s_{x2} & 0 & 0 \\ 0 & s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_{x1} & 0 & 0 \\ 0 & s_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{x1} \cdot s_{x2} & 0 & 0 \\ 0 & s_{y1} \cdot s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-29)$$

또는

$$\mathbf{S}(s_{x2}, s_{y2}) \cdot \mathbf{S}(s_{x1}, s_{y1}) = \mathbf{S}(s_{x1} \cdot s_{x2}, s_{y1} \cdot s_{y2}) \tag{5-30}$$

이 결과행렬은 련속된 비례변환연산들의 곱하기라는것을 보여 준다. 즉 물체의 크기를 련속 두번 3배하면 마지막크기는 본래의 9배가 된다.

일반회전축점주위의 회전

자리표원점에 대하여 회전하는 물체에 대한 회전함수만을 제공하는 도형처리프로그람에서는 다음과 같은 평행이동-회전 -평행이동연산을 진행하여야 임의로 선택되는 회전축점 (x_r , y_r)에 대한 회전을 발생시킬수 있다.

1. 회전축점의 위치가 자리표원점으로 되도록 물체를 평행이동시킨다.
2. 자리표원점에 대하여 물체를 회전시킨다.
3. 회전축점이 자기의 본래 위치로 돌아 가도록 물체를 평행이동시킨다.

이 변환순서를 그림 5-9에서 설명한다. 이 순서렬에 대한 합성변환행렬은

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r \sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r \sin\theta \\ 0 & 0 & 1 \end{bmatrix} \tag{5-31}$$

결합에 의하여 얻어 진다. 이것은

$$\mathbf{T}(x_r, y_r) \cdot \mathbf{R}(\theta) \cdot \mathbf{T}(-x_r, -y_r) = \mathbf{R}(x_r, y_r, \theta) \tag{5-32}$$

형식으로 표현할수 있다. 여기서 $\mathbf{T}(-x_r, -y_r) = \mathbf{T}^{-1}(x_r, y_r)$ 이다. 일반적으로 회전함수는 식 5-31의 회전변환행렬을 자동적으로 발생시키도록 회전각뿐아니라 회전축점의 자리표에 대한 파라메터들도 받아 들이도록 설정될수 있다.

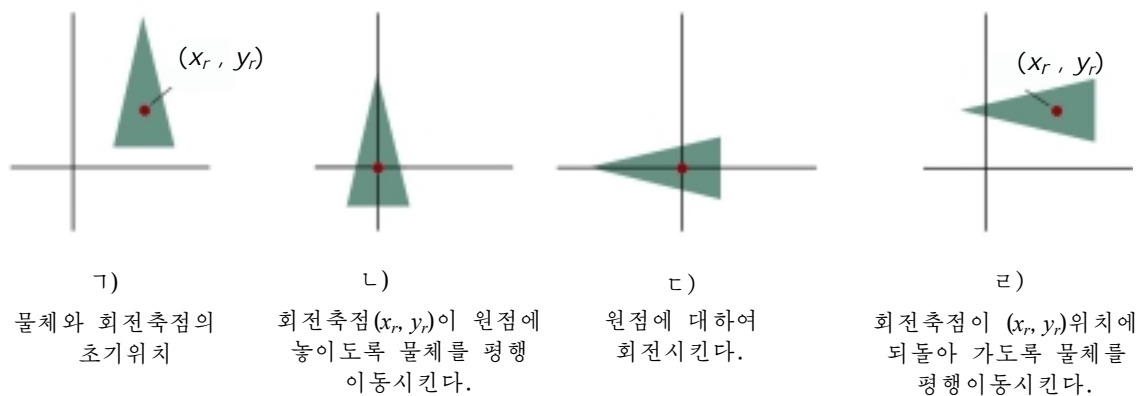



그림 5-9. 변환식 5-19의 회전행렬 $\mathbf{R}(\theta)$를 리용하여 지적된 회전축점에 대하여 물체를 회전시킬 때의 변환순서

일반고정점에 대한 비례변환

그림 5-10에서는 자리표원점에 대한 비례변환만 할수 있는 비례변환함수를 리용하여 임의로 선택된 고정점위치 (x_f, y_f)에 대한 비례변환을 얻는 변환순서를 보여 주었다.

1. 고정점이 자리표원점과 일치하도록 물체를 평행이동시킨다.
2. 자리표원점에 대하여 물체를 비례변환한다.
3. 물체를 본래 위치로 돌려 보내기 위하여 걸음 1의 역평행이동을 진행한다.

이 세개의 조작에 대한 행렬들을 결합하면 요구되는 비례변환행렬을 얻는다.

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_f(1-s_x) \\ 0 & s_y & y_f(1-s_y) \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-33)$$

또는

$$\mathbf{T}(x_f, y_f) \cdot \mathbf{S}(s_x, s_y) \cdot \mathbf{T}(-x_f, -y_f) = \mathbf{S}(x_f, y_f, s_x, s_y) \qquad (5-34)$$

이 변환은 고정점자리표를 받아 들이는 비례변환함수를 제공하고 있는 체계들에서 자동적으로 수행된다.

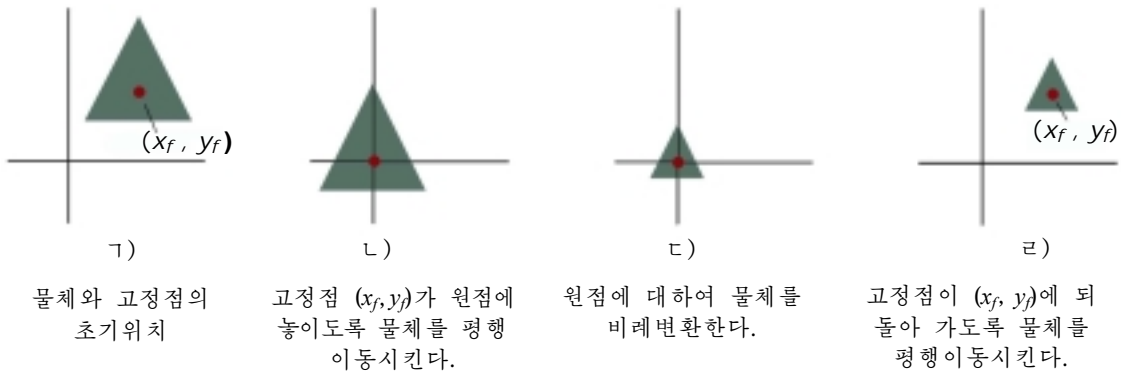



그림 5-10. 변환식 5-21의 비례변환행렬 $\mathbf{S}(s_x, s_y)$를 리용하여 지적된 고정점 위치에 대하여 물체를 비례변환하는 변환순서

일반비례변환방향

파라메터 s_x와 s_y는 물체를 x와 y방향으로 비례변환시킨다. 비례변환을 적용하기전에 요구되는 비례변환방향을 자리표축과 일치시키도록 물체를 회전시키면 물체를 다른 방향으로 비례변환할수 있다.

파라메터 s_1 및 s_2에 의하여 지적되는 값을 가지는 비례결수들을 그림 5-11에 보여 준 방향으로 적용한다고 하자. 물체의 방향을 변화시킴이 없이 비례변환을 수행하기 위해서는 먼저 s_1 및 s_2의 방향이 x 및 y축과 일치하도록 회전을 진행한다. 다음에 비례변환을 적용하고 점들을 다시 자기의 본래 방향에로 되돌려 보내는 반대방향회전이 뒤따른다. 이 세 변환적의 결과로 이루어 지는 합성행렬은

$$\mathbf{R}^{-1}(\theta) \cdot \mathbf{S}(s_1, s_2) \cdot \mathbf{R}(\theta)$$

$$= \begin{bmatrix} s_1\cos^2\theta + s_2\sin^2\theta & (s_2 - s_1)\cos\theta\sin\theta & 0 \\ (s_2 - s_1)\cos\theta\sin\theta & s_1\sin^2\theta + s_2\cos^2\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-35)$$

이 비례변환의 실례로서 그림 5-12 ㄱ의 단위바른4각형을 (0,0)부터 (1,1)까지의 대각선을 따라 잡아 당겨 평행4변형으로 변환하자. 대각선을 y축에 회전시키고 그의 길이를 변환파라메터 $\theta=45°$, $s_1=1$, $s_2=2$로 2배 한다.

식 5-35에서는 비례변환이 원점에 대하여 수행된다고 가정하였다. 이 비례변환조작을 한걸음 더 전진시켜 행렬을 평행이동연산자와 결합시킴으로써 합성행렬이 비례변환의 고정점위치지적에 대한 파라메터를 포함하게 할수 있다.

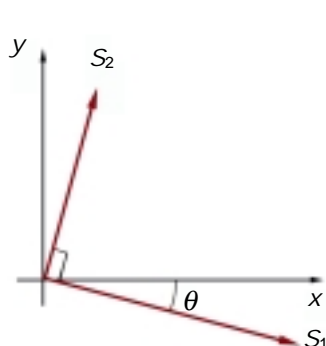



그림 5-11. 비례변환파라메터 s_1 및 s_2는 각변위 θ에 의하여 정의되는 직각방향으로 적용된다.

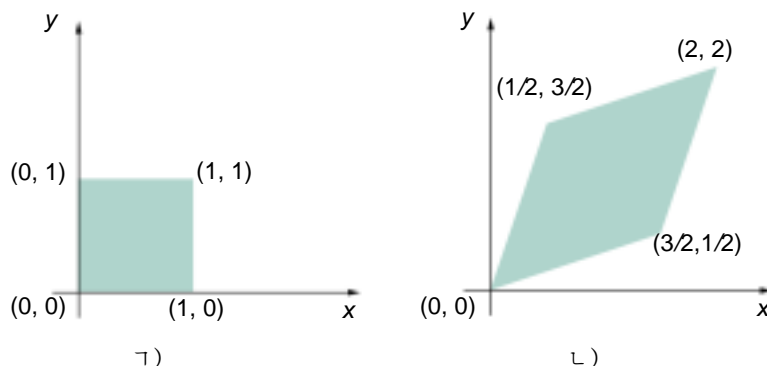



그림 5-12. 바른4각형(ㄱ)을 식 5-35의 합성변환행렬 $s_1=1$, $s_2=2$, $\theta=45°$를 리용하여 평행4변형(ㄴ)으로 변환한다.

련결성

행렬곱하기에서는 결합칙이 성립한다. 임의의 세개 행렬 **A**, **B**, **C**에 대한 행렬적 **A**·**B**·**C**는 먼저 **A**와 **B**를 곱하거나 먼저 **B**와 **C**를 곱하는 방법으로 진행할수 있다.

$$\mathbf{A}\cdot\mathbf{B}\cdot\mathbf{C}=(\mathbf{A}\cdot\mathbf{B})\cdot\mathbf{C}=\mathbf{A}\cdot(\mathbf{B}\cdot\mathbf{C}) \tag{5-36}$$

따라서 행렬적을 왼쪽에서 오른쪽으로 또는 오른쪽에서 왼쪽으로 결합하여 계산할수 있다.

다른 한편 변환행렬의 적은 교환이 성립되지 않을수 있다. 일반적으로 행렬적 **A**·**B**는 **B**·**C**와 같지 않다. 이것은 물체를 평행이동 및 회전시키려 할 때 합성행렬이 계산되는 순서에 대하여 주의하여야 한다는것을 의미한다(그림 5-13). 모두 같은 종류의 변환의 순서렬인 경우와 같이 일부 특수한 경우들에서는 변환행렬의 곱하기가 교환적이다. 실례로 두개의 련속된 회전은 어느 순서로 수행하여도 마지막위치는 같게 된다. 이런 교환특성은 두개의 련속된 평행이동 또는 두개의 련속된 비례변환에 대해서도 성립한다. 회전과 동형비례변환($s_x=s_y$)에서도 교환이 성립한다.

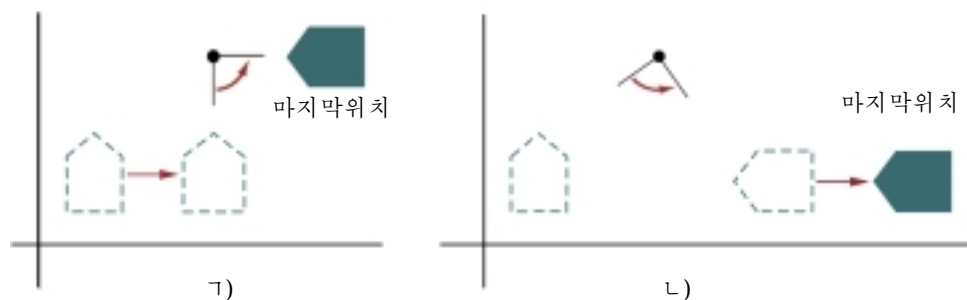



그림 5-13. 변환순서렬의 수행순서를 바꾸면 변환된 물체의 위치에 영향을 미칠수 있다(ㄱ에서는 물체가 먼저 평행이동되고 다음에 회전된다. ㄴ에서는 물체가 먼저 회전되고 다음에 평행이동된다.).

일반합성변환 및 계산효률

평행이동, 회전, 비례변환의 결합을 표현하는 일반2차원변환은

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} rs_{xx} & rs_{xy} & trs_x \\ rs_{yx} & rs_{yy} & trs_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (5-37)$$

와 같이 표현할수 있다. 4개의 요소 rs_{ij}는 회전각과 비례결수만을 포함하는 변환에서 곱하기회전-비례변환항들이다. 요소 trs_x와 trs_y는 평행이동거리, 회전축점 및 고정점자리표, 회전각 및 비례변환파라메터들과 결합된 이행항이다. 실례로 물체가 그의 중심자리표 $(x_c,\ y_c)$에 대하여 비례변환 및 회전되고 다음에 평행이동된다면 합성변환행렬의 요소들의 값은

$$\mathbf{T}(t_x,t_y)\cdot\mathbf{R}(x_c,y_c,\theta)\cdot\mathbf{S}(x_c,y_c,s_x,s_y)$$

$$= \begin{bmatrix} s_x\cos\theta & -s_y\sin\theta & x_c(1-s_x\cos\theta)+y_c s_y\sin\theta+t_x \\ s_x\sin\theta & s_y\cos\theta & y_c(1-s_y\cos\theta)-x_c s_x\sin\theta+t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-38)$$

비록 행렬방정식 5-37은 9번의 곱하기와 6번의 더하기를 요구하지만 변환된 자리표에 대한 양함수적인 계산은

$$x' = x\cdot rs_{xx} + y\cdot rs_{xy} + trs_x, \qquad y' = x\cdot rs_{yx} + y\cdot rs_{yy} + trs_y \qquad (5-39)$$

이다. 이리하여 자리표위치를 변환하는데는 실제적으로 4번의 곱하기와 4번의 더하기만 수행하면 된다. 이것은 개별적인 행렬들을 결합시킨 합성행렬의 요소들이 계산된후에 임의의 변환순서렬에 대하여 요구되는 최대계산량이다. 결합이 없는 경우에는 개별적인 변환들이 한번에 하나씩 적용되게 되며 계산량은 상당히 증가되게 된다. 따라서 변환조작에 대한 능률적인 실현은 변환행렬들을 형식화하고 임의의 변환순서렬을 결합한 다음 식 5-39를 리용하여 변환자리표들을 계산하는것이다. 병렬체계에서도 식 5-37의 합성변환행렬에 의한 직접 행렬곱하기가 효률적이다.

평행이동과 회전만을 포함하는 일반강체변환행렬은

$$\begin{bmatrix} r_{xx} & y_{xy} & tr_x \\ r_{yx} & y_{yy} & tr_y \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-40)$$

의 형식으로 표현할수 있다. 여기서 4개의 요소 r_{ij}는 곱하기적인 회전항이고 요소 trs_x와 tr_y는 평행이동항이다. 강체의 자리표위치에서의 변화를 때때로 강체운동변환이라고도 한다. 자리표위치들사이의 모든 각과 거리는 변환에 의하여 변화되지 않는다. 게다가 행렬식 5-40은 그의 왼쪽 우 2×2부분행렬이 직교행렬이라는 특성을 가진다. 이것은 부분행렬의 매행을 벡토르로 고찰할 때 두개의 벡토르 (r_{xx},r_{xy})와 (r_{yx},r_{yy})는 단위벡토르들의 직교모임을 형성한다는것을 의미한다. 매 벡토르는 단위길이

$$r_{xx}^2 + r_{xy}^2 = r_{yx}^2 + r_{yy}^2 = 1 \qquad (5-41)$$

을 가지며 벡토르들은 수직이다(그것들의 스칼라적은 0이다.).

$$r_{xx}r_{yx} + r_{xy}r_{yy} = 0 \qquad (5-42)$$

따라서 이 단위벡토르들을 회전부분행렬에 의하여 변환하면 (r_{xx},r_{xy})는 x축단위벡토르로 변환되고 (r_{yx},r_{yy})는 자리표계의 y축단위벡토르로 변환된다. 즉

$$\begin{bmatrix} r_{xx} & r_{xy} & 0 \\ r_{yx} & r_{yy} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{xx} \\ r_{xy} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \qquad (5-43)$$

$$\begin{bmatrix} r_{xx} & r_{xy} & 0 \\ r_{yx} & r_{yy} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} r_{yx} \\ r_{yy} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \qquad (5-44)$$

실례로 다음의 강체변환은 먼저 물체를 회전축점 (x_r, y_r)에 대하여 각 θ만큼 회전시키고 다음 평행이동시킨다.

$$\mathbf{T}(t_x, t_y) \cdot \mathbf{R}(x_r, y_r, \theta) = \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r \sin\theta + t_x \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r \sin\theta + t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-45)$$

여기서 왼쪽 우 2×2 부분행렬에서 직교단위벡토르들은 $(\cos\theta, -\sin\theta)$ 및 $(\sin\theta, \cos\theta)$이며

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta \\ -\sin\theta \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \qquad (5-46)$$

이다. 류사하게 단위벡토르 $(\sin\theta, \cos\theta)$는 식 5-46의 변환행렬에 의하여 y방향의 단위벡토르 $(0,1)$로 변환된다.

회전행렬의 직교특성은 물체를 어떤 위치로 놓는데 필요한 회전각크기가 아니라 물체의 마지막 방향을 아는 경우 회전행렬을 만드는데서 쓸모 있다. 물체에 요구되는 방향은 장면안에서 주어 진 물체를 어디에 어떻게 놓는가에 따라 결정할수 있다.

그림 5-14에서는 단위방향벡토르 $\mathbf{u}'$와 $\mathbf{v}'$에 일직선되게 하여야 할 물체를 보여 주었다. 그림 5-14 ㄱ에서 보여 준바와 같이 물체의 본래 방향이 자리표축과 일직선을 이룬다고 가정하면 요구되는 변환은 $\mathbf{u}'$ 의 요소들을 회전행렬의 첫번째 행에 할당하고 $\mathbf{v}'$ 의 요소들을 두번째 행에 할당하면 얻어 진다. 이것은 마지막방향벡토르를 알고 있을 때 국부(또는 《물체》)자리표계에서 회전변환행렬을 얻는 편리한 방법으로 될수 있다. 류사한 변환은 물체서술의 한 자리표계로부터 다른 자리표계로의 변환이다. 5절에서 이 자리표변환을 수행하기 위하여 변환행렬을 어떻게 설정하는가 하는데 대하여 고찰한다.

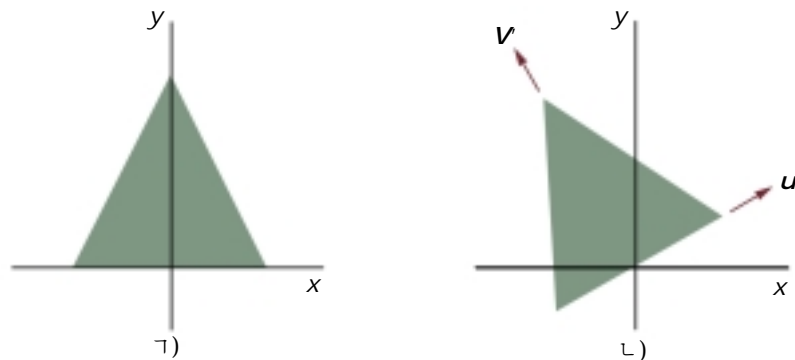



그림 5-14. 위치 ㄱ로부터 위치 ㄴ에로의 물체회전에 대한 회전행렬은 본래 방향에 대한 단위방향벡토르 $\mathbf{u}'$와 $\mathbf{v}'$의 값에 의하여 만들수 있다.

회전계산은 변환되는 매점에 대하여 삼각계산과 곱하기를 여러번 요구하기때문에 회전변환에서는 계산효률이 고찰되여야 할 중요한 문제로 될수 있다. 수많은 반복변환과 작은 걸음각을 포함하는

동화 및 다른 응용들에서는 합성변환식에서 계산을 줄이기 위하여 근사식과 대화식계산을 리용할수 있다. 회전각이 작을 때는 삼각함수를 그의 제곱합렬전개의 첫 몇개 항에 기초한 근사값으로 교체할 수 있다. 충분히 작은 각(10°보다 작은)에 대한 cos θ는 약 1이며 sin θ는 라디안으로 θ값에 대단히 가까운 값을 가진다. 실례로 원점에 대하여 작은 걸음각으로 회전하고 있다면 cos θ는 1로 설정할수 있으며 매 걸음에서의 변환계산은 매 회전될 자리표들의 모임에 대하여 두번의 곱하기와 두번의 더하기로 줄일수 있다.

$$x' = x - y\sin\theta, \qquad y' = x\sin\theta + y \qquad (5-47)$$

여기서 sin θ는 걸음각이 변하지 않는다는 가정하에서 모든 걸음들에 대하여 한번만 계산한다. 매 걸음에서의 이 근사에 의한 오차는 걸음각이 작을수록 작다. 그러나 매우 작은 걸음각이라 하여도 많은 걸음들에서 축적되면 오차가 아주 크게 될수 있다. 축적오차는 매 걸음에서 x'와 y'에서의 오차를 추정하고 오차의 축적이 너무 커질 때 물체의 위치를 재설정하는 방법으로 조종할수 있다.

합성변환에는 자주 역행렬계산이 들어 간다. 실례로 일반비례변환의 방향과 반사와 쏠림(5절)에 대한 변환순서렬은 역회전요소에 의하여 서술할수 있다. 이미 지적한것처럼 기하학적기본변환에 대한 역행렬표현은 간단한 절차로 얻을수 있다. 역평행이동행렬은 평행이동거리의 부호를 바꾸면 얻어 지며 역회전행렬은 행렬전위(또는 시누스항의 부호를 변화시키는것)에 의하여 얻어 진다. 이 연산들은 역행렬의 직접적인 계산보다도 훨씬 더 간단하다.

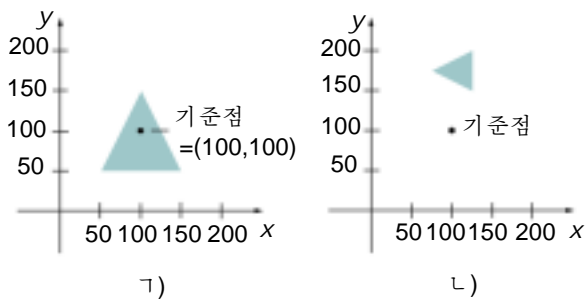



그림 5-15. 다각형 ㄱ가 다음 절차의 합성조작에 의해 ㄴ로 변환된다.

합성변환의 실현을 다음의 절차에 주었다. 행렬 **M**은 단위행렬로 초기화된다. 매개 개별적인 변환이 지적되면 그것은 총 변환행렬 **M**에 결합된다. 모든 변환이 다 지적되였으면 이 합성변환이 주어 진 물체에 적용된다. 이 실례에서는 다각형을 주어 진 기준점에 대하여 비례변환 및 회전시키고 다음에 평행이동시킨다. 그림 5-15는 이 순서에 의하여 변환되는 다각형의 본래 위치와 마지막위치를 보여 준다.

```
#include <math.h>
#include "graphics.h"

typedef float Matrix3x3[3][3];
Matrix3x3 theMatrix;
void matrix3x3SetIdentity (Matrix3x3 m)
{
  int i , j ;

  for (i=0; i<3; i++) for (j=0; j<3; j++) m[i][j] = (i == j ) ;
}

/* Multiplies matrix a times b, putting result in b */
void matrix3x3PreMultiply (Matrix3x3 a, Matrix3x3 b)
{
  int r,c;
```

```
  Matrix3x3 tmp;

  for (r = 0; r < 3; r++)
   for (c = 0; c < 3; C++)
     tmp [r] [c] =
      a[r][0]*b[0][c]+a[r][1]*b[1][c]+a[r][2]*b[2][c] ;

  for (r = 0; r < 3; r++)
   for (c = 0; c < 3; C++)
     b[r][c] = tmp[r][c] ;
}

void translate2 (int tx, int ty)
{
  Matrix3x3 m;

  matrix3x3SetIdentity (m) ;
  m[0][2] = tx;
  m[l][2] = ty;
  matrix3x3PreMultiply (m, theMatrix) ;
}

void scale2 (float sx, float sy, wcPt2 refpt)
{
  Matrix3x3 m;

  matrix3x3SetIdentity (m) ;
  m[0][0] = sx;
  m[0][2] = (1 - sx) * refpt.x;
  m[l][1] = sy;
  m[l][2] = (1 - sy) * refpt.y;
  matrix3x3PreMultiply (m, theMatrix) ;
}

void rotate2 (float a, wcPt2 refPt)
{
  Matrix3x3 m;

  matrix3x3SetIdentity (m) ;
  a = pToRadians (a) ;
  m[0][0] = cosf (a) ;
  m[0][1] = -sinf (a) ;
  m[0][2] = refPt.x * (1 - cosf (a)) + refPt.y * sinf (a) ;
  m[l][0] = sinf (a) ;
  m[l][1] = cosf (a) ;
  m[l][2] = refPt.y * (1 - cosf (a)) - refPt.x * sinf (a);
  matrix3x3PreMultiply (m, theMatrix) ;
```

```
}

void transformPoints2 (int npts, wcPt2 *pts)
{
  int k;
  float tmp;

  for (k = 0; k < npts; k++) {
   tmp = theMatrix[0][0] * pts[k].x + theMatrix[0][1] *
     pts[k].y + theMatrix[0][2];
   pts[k].y = theMatrix[1][0] * pts[k].x + theMatrix[1][1] *
     pts[k].y + theMatrix[1][2];
   pts[k].x = tmp;
  }
}

void main (int argc, char ** argv)
{
  wcPt2 pts[3] = { 50.0, 50.0, 150.0, 50.0, 100.0, 150.0};
  wcPt2 refPt = {100.0, 100.0};
  long windowID = openGraphics (*argv, 200, 350);

  setBackground (WHITE) ;
  setColor (BLUE) ;
  pFillArea (3, pts);
  matrix3x3SetIdentity (theMatrix) ;
  scale2 (0.5, 0.5, refPt);
  rotate2 (90.0, refPt);
  translate2 (0, 150) ;
  transformPoints2 (3, pts);
  pFillArea (3,pts) ;
  sleep (10);
  closeGraphics (windowID) ;
}
```

4절. 기타 변환

평행이동, 회전, 비례변환과 같은 기본변환들은 대부분 도형처리프로그람들에 들어 있다. 일부 프로그람들은 일정한 응용에서 쓸수 있는 몇가지 추가적인 변환들도 가지고 있다. 이러한 변환이 반사와 쏠림이다.

반사

반사는 물체의 거울상을 만드는 변환이다. 2차원반사에 의한 거울상은 물체를 반사축에 대하여 180° 회전시킴으로써 얻어 진다. 반사축은 xy평면에서 또는 xy평면에 수직으로 선택할수 있다. 반사

축이 xy평면의 선일 때 이 축에 대한 회전경로는 xy평면에 수직인 면에 있다. xy평면에 수직인 반사축에 대한 회전경로는 xy평면에 있다. 다음의것들은 대표적인 몇가지 반사의 실례들이다.

선 $y=0$ 즉 x축에 대한 반사는 변환행렬

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-48}$$

에 의하여 수행된다. 이 변환은 x값은 그대로 보존하고 자리표위치의 y값은 뒤집는다. x축에 대하여 물체가 반사된후 물체의 결과적인 방향을 그림 5-16에 보여 주었다. 이 반사에 대한 회전변환경로를 상상하면 평탄한 물체를 xy평면밖으로 움직이되 3차원공간에서 x축에 대하여 180° 회전시켜 xy평면의 x축의 반대쪽에 뒤집어 놓은것으로 생각할수 있다.

y축에 대한 반사는 x자리표를 뒤집고 y자리표는 그대로 보존한다. 이 변환에 대한 행렬은

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-49}$$

이다. 그림 5-17에서는 선 $x=0$에 대하여 반사된 물체에서의 자리표위치의 변화를 보여 주었다. 이 경우는 3차원공간에서 y축에 대하여 180° 회전시킨것과 같다.

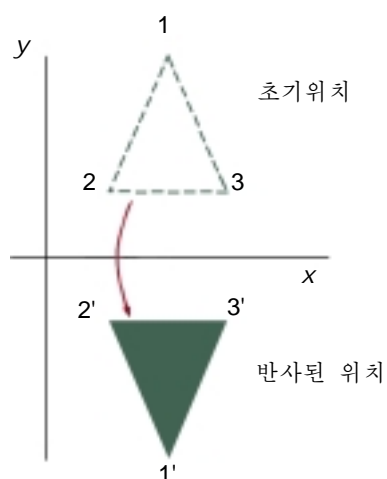



그림 5-16. x축에 대한 물체의 반사

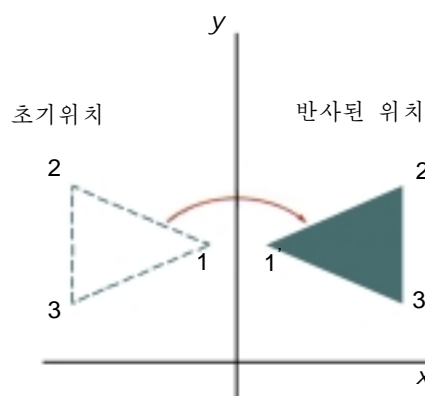



그림 5-17. y축에 대한 물체의 반사

xy평면에 수직이고 자리표원점을 통과하는 축에 대하여 반사시키면 점의 x 및 y자리표를 다같이 뒤집는다. 자리표원점에 대한 반사라고 하는 이 변환은 행렬표현

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-50}$$

을 가진다. 원점에 대한 반사의 실례를 그림 5-18에 보여 주었다. 반사행렬 5-50은 θ=180° 인 회전행렬 R(θ)이다. 물체를 xy평면에서 원점에 대하여 간단히 반바퀴회전시키고 있다.

반사행렬 5-50 은 xy평면의 임의의 반사점에 대하여 일반화할수 있다(그림 5-19). 이 반사는 반사점을 회전축점처럼 리용하는 xy평면에서의 180° 회전과 같다.

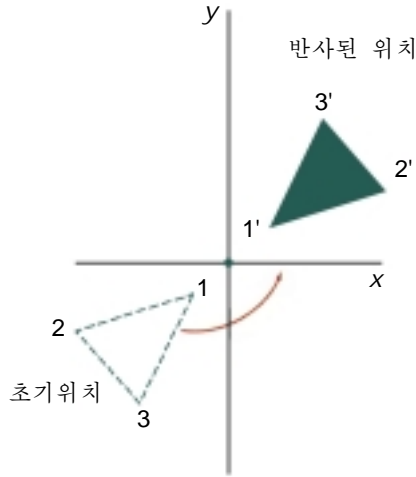



그림 5-18. xy평면에 수직이고 자리표원점을 통과하는 축에 대한 물체의 반사

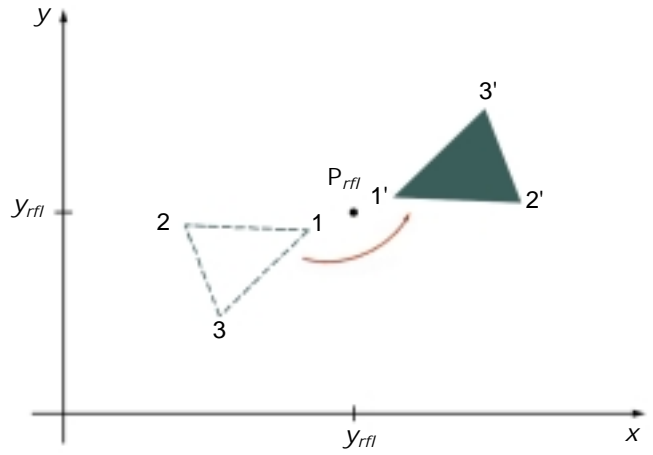



그림 5-19. xy평면에 수직이고 점 P_{rfl}을 통과하는 축에 대한 물체의 반사

만약 반사축을 대각선 $y=x$ 로 선택하면(그림 5-20) 반사행렬은

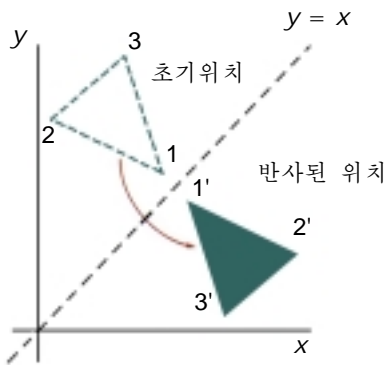
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-51}$$

이다. 이 행렬은 회전 및 자리표축반사행렬들의 순서렬을 결합하여 유도할수 있다. 한가지 가능한 순서를 그림 5-21에 보여주었다. 여기서는 먼저 시계바늘방향으로 45° 회전시킨다. 이것은 선 $y=x$를 x축에로 회전시킨다. 다음 x축에 대하여 반사시킨다. 마지막걸음은 시계바늘반대방향으로 45° 회전시켜 선 $y=x$를 자기의 본래위치에로 되돌려 보내는것이다. 변환의 다른 한가지 순서는 먼저 물체를 x축에 대하여 반사시키고 다음 시계바늘반대방향으로 90° 회전시키는것이다.

그림 5-20. 선 $y=x$ 에 대한 물체의 반사

대각선 $y=-x$에 대한 반사의 변환행렬을 얻기 위하여는 다음의 변환순서에 의해 행렬들을 결합한다. (1) 시계바늘방향으로 45° 회전, (2) y축에 대한 반사, (3) 시계바늘 반대방향으로 45° 회전시킨다. 결과적인 변환행렬은

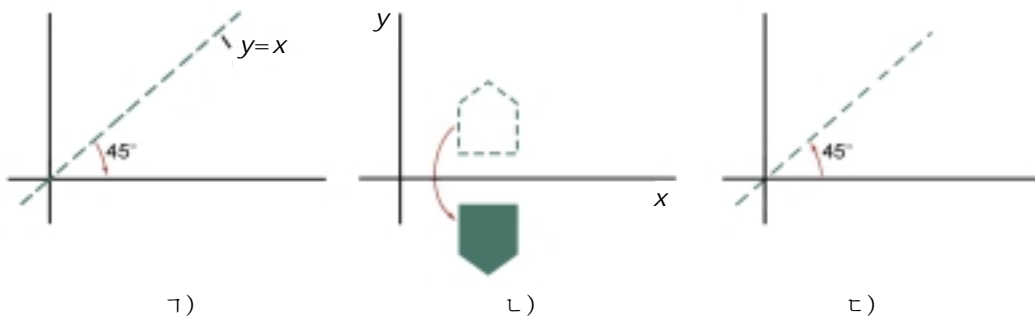



그림 5-21. 선 $y=x$ 에 대한 반사를 얻기 위한 변환순서
ㄱ-시계바늘방향으로 45° 회전, ㄴ-x 축에 대한 반사,
ㄷ-시계바늘반대방향으로 45° 회전

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-52)$$

이다. 그림 5-22에서는 이 반사행렬에 의하여 변환되는 물체에 대한 초기위치와 마지막위치를 보여 주었다.

xy평면에서 임의의 선 $y=mx+b$에 대한 반사는 평행이동-회전-반사 변환들의 결합에 의해 수행할수 있다. 일반적으로 먼저 선이 원점을 통과하도록 평행이동시킨다. 다음에 선을 자리표축들중 하나에로 회전시키고 그 축에 대하여 반사시킨다. 마지막으로 역회전과 평행이동변환에 의하여 선을 자기의 본래 위치에로 돌려 보낸다.

자리표축 또는 자리표원점에 대한 반사를 부의 비례결수를 가지는 비례변환과 같이 실현할수 있다. 또한 반사행렬의 원소를 ±1이 아닌 다른 값으로도 설정할수 있다. 크기가 1보다 큰 값은 거울상을 반사축으로부터 더 멀리 밀며 크기가 1보다 작은 값은 거울상을 반사축에 더 가깝게 가져 온다.

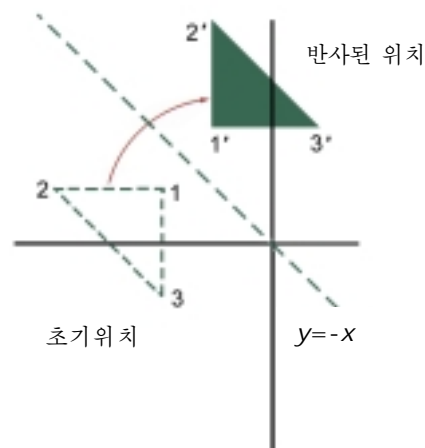



그림 5-22. 선 $y=x$ 에 대한 반사

쏠림

물체의 형태를 마치 서로 미끌어 지는 내부층들로 이루어 진것처럼 찌그러뜨리는 변환을 쏠림(shear)변환이라고 한다. 대표적인 두개의 쏠림변환은 자리표 x값을 미는것과 y값을 미는것이다. x축에 대한 x방향쏠림은 변환행렬

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (5-53)$$

에 의하여 만들어 진다. 이것은 자리표위치를

$$x' = x + sh_x \cdot y, \qquad y' = y \qquad (5-54)$$

와 같이 변환한다. 쏠림파라메터 sh_x에는 임의의 실수값이 할당될수 있다. 그러면 자리표위치 (x, y)는 x축$(y=0)$으로부터의 자기의 거리$(y=0)$에 비례하는 량만큼 수평으로 밀려 진다. 실례로 sh_x를 2로 설정하면 그림 5-23의 바른4각형을 평행4변형으로 변화시킨다. sh_x에 대한 부의 값은 자리표위치를 왼쪽으로 민다.

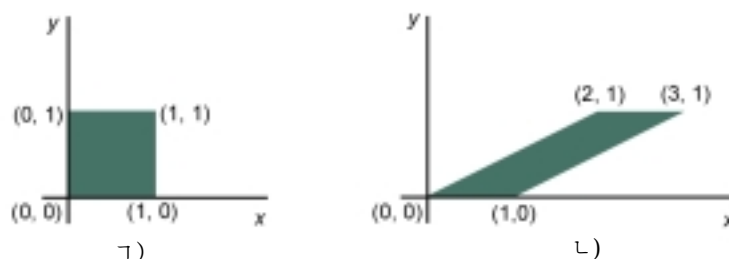



그림 5-23. 단위바른4각형 ㄱ를 $sh_x=2$인 식 5-53의 x방향 쏠림행렬을 리용하여 평행4변형 ㄴ로 변환한다.

다른 기준선에 대한 x방향쏠림은

$$\begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{\text{ref}} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-55}$$

에 의하여 발생시킬수 있다. 자리표위치들은

$$x' = x + sh_x(y - y_{\text{ref}}),\ \ y' = y \tag{5-56}$$

와 같이 변환된다. 선 y_{ref} =−1에 대하여 쏠림파라메터값 1/2을 가지고 진행한 쏠림변환의 실례를 그림 5-24에 주었다.

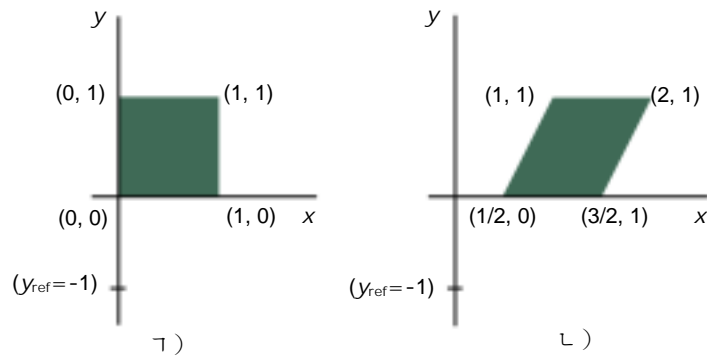



그림 5-24. 단위바른4각형 ㄱ은 식 5-55의 쏠림행렬에서 sh_x=1/2, y_{ref}=-1에 의하여 밀려 평행4변형 ㄴ로 변환된다.

선 $x = y_{ref}$에 대한 y 방향쏠림은 변환행렬

$$\begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & -sh_y \cdot x_{\text{ref}} \\ 0 & 0 & 1 \end{bmatrix} \tag{5-57}$$

에 의하여 발생된다. 이것은 변환된 자리표위치

$$x' = x,\ \ y' = sh_y(x - x_{\text{ref}}) + y \tag{5-58}$$

를 발생시킨다. 이 변환은 자리표위치를 기준선 $x = x_{ref}$로부터의 거리에 비례하는 크기만큼 수직으로 민다. 그림 5-25는 sh_y=1/2, x_{ref} =−1에 의한 바른4각형의 평행4변형에로의 변환을 설명한다.

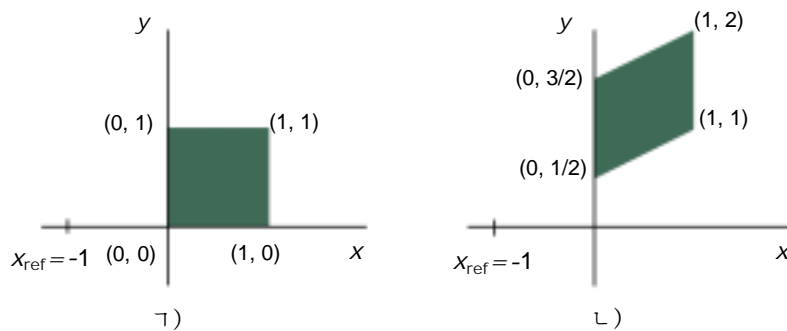



그림 5-25. 단위바른4각형 ㄱ는 식 5-57의 쏠림변환을 리용하여 y방향에서 파라메터값 sh_y=1/2, x_{ref}=−1에 의해 밀려 평행4변형 ㄴ로 변환된다.

쏠림조작은 기본변환들의 순서렬로 표현할수 있다. 실례로 x방향쏠림행렬 5-53은 그림 5-23의 단위바른4각형을 그의 대각선을 따라 비례변환하고 한편 x축에 평행인 변의 본래 길이와 방향을 유지하는 회전 및 비례변환행렬들의 련속을 포함하는 합성변환으로 쓸수 있다. 쏠림기준선에 대한 물체위치의 밀기는 평행이동과 등가이다.

5절. 자리표계들사이의 변환

도형처리응용에서는 흔히 물체서술을 한 자리표계로부터 다른 자리표계로 변환하게 된다. 때로는 대칭성의 우점을 살리는 비직각자리표계로도 물체들을 서술한다.이런 자리표계에서의 자리표서술을 현시하려면 장치의 직각자리표로 변환하여야 한다. 2차원의 비직각자리표계의 몇가지 실례는 극자리표, 타원자리표, 포물선자리표계이다. 어떤 경우에는 두 직각자리표계사이에서 변환하는것도 필요하다. 모형화 및 설계응용에서 개별적인 물체들은 자기자체의 국부적인 직각자리표계로 정의될수 있는데 이 국부자리표는 그다음 물체의 위치를 전체적인 장면자리표계안에서 규정하려면 변환되여야 한다. 실례로 사무실설계관리프로그람에서는 의자, 책상 그리고 방바닥에 놓을수 있는 기타 가구비품들을 개별적인 자리표계에서 서술하고 그것들을 여러 위치에 다중복사한다. 어떤 응용들에서는 단순히 자리표계의 방향만 다시 정하여 장면을 현시할수 있다. 직각자리표계와 몇가지 대표적인 비직각자링표계들사이의 관계를 부록 1에 주었다. 이 절에서는 두개의 직각자리표계사이의 변환을 고찰한다.

그림 5-26. xy 직각자리표계안에서 방향각이 θ이고 원점이 $(x_0\ ,\ y_0)$에 놓이는 $x'y'$직각자리표계

그림 5-26에서는 자리표원점이 $(0,0)$과 $(x_0\ ,\ y_0)$에 있고 x 및 x'축사이의 방향각이 θ인 두개의 직각자리표계를 보여 주었다. 물체서술을 xy자리표로부터 $x'y'$자리표에로 변환하기 위하여서는 $x'y'$축들을 xy축에 덧놓는 변환을 진행하여야 한다. 이것은 다음의 두걸음으로 진행된다.

1. $x'y'$계의 원점 $(x_0\ ,y_0)$이 xy계의 원점에 놓이도록 평행이동시킨다.
2. x'축을 x 축에로 회전시킨다.

자리표원점의 평행이동은 행렬연산

$$T(-x_0,-y_0)=\begin{bmatrix}1 & 0 & -x_0\\ 0 & 1 & -y_0\\ 0 & 0 & 1\end{bmatrix} \qquad (5-59)$$

에 의해 표현된다. 평행이동조작후 두 계의 방향은 그림 5-27과 같이 된다. 다음에 두 계의 축들이 일치되도록 하기 위하여 시계바늘방향의 회전을 진행한다.

$$R(-\theta)=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \qquad (5-60)$$

이 두 변환행렬을 결합하면 물체서술을 xy계로부터 $x'y'$계에로 변환하는 완전한 합성행렬이 얻어진다.

$$\mathbf{M}_{xy,x'y'}=\mathbf{R}(-\theta)\cdot\mathbf{T}(-x_0,-y_0) \qquad (5-61)$$

두번째 자리표계의 방향을 주는 또 하나의 방법은 그림 5-28에 보여 준바와 같이 정의 y'축방향을 가리키는 벡토르 $\mathbf{V}$를 지적하는것이다. 벡토르 $\mathbf{V}$는 xy계의 원점에 대한 xy자리표계의 점으로 지적한다. 그러면 y'방향의 단위벡토르는

$$\mathbf{v} = \frac{\mathbf{V}}{|\mathbf{V}|} = (v_x,\ v_y) \tag{5-62}$$

와 같이 얻을수 있다. 그리고 x'축단위벡토르 $\mathbf{u}$는 $\mathbf{v}$를 시계바늘방향으로 90° 회전시켜 얻는다.

$$\begin{aligned}\mathbf{u} &= (v_y, -v_x)\\ &= (u_x, u_y)\end{aligned} \tag{5-63}$$

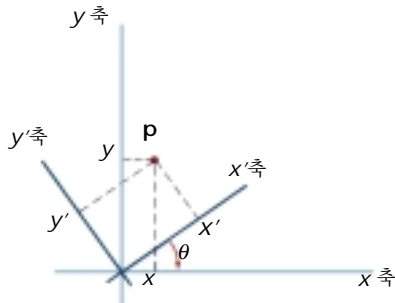



그림 5-27. $x'y'$계의 원점을 xy계의 자리표 원점에로 평행이동시킨후의 그림 5-26 에 보여 준 자리표계들의 위치

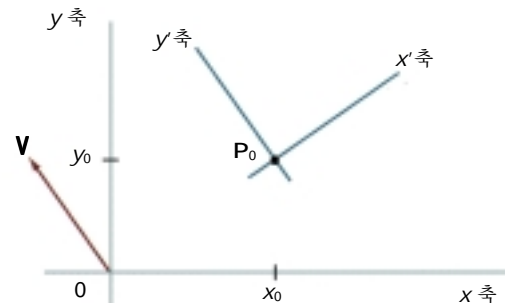



그림 5-28. 원점이 $\mathrm{P}_0 = (x_0, y_0)$에 있고 y'축이 벡토르 $\mathbf{V}$에 평행인 $x'y'$직각자리표계

3 절에서 임의의 회전행렬의 원소들은 직교하는 단위벡토르들의 모임의 원소들로 표현할수 있다는것을 보았다. 따라서 $x'y'$계를 xy계와 일치하도록 회전시키는 행렬은

$$R = \begin{bmatrix} u_x & u_y & 0 \\ v_x & v_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-64}$$

와 같이 쓸수 있다. 실례로 y'축의 방향을 $\mathbf{V} = (-1, 0)$과 같이 선정하였다고 하자. 그러면 x'축은 정의 y방향에 놓이게 되며 회전변환행렬은

$$\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

이다. 등가적으로 이 회전행렬은 식 5-60으로부터 방향각을 $\theta = 90°$로 설정하여도 얻을수 있다.

대화식응용에서는 위치 P_0에 대한 $\mathbf{V}$의 방향을 선정하는것이 xy자리표원점에 대한 지적보다 더 편리할수 있다. 이때 단위벡토르 $\mathbf{u}$와 $\mathbf{v}$는 그림 5-29에 보여 준바와 같이 방향이 결정된다. 이때 $\mathbf{v}$의 성분들은

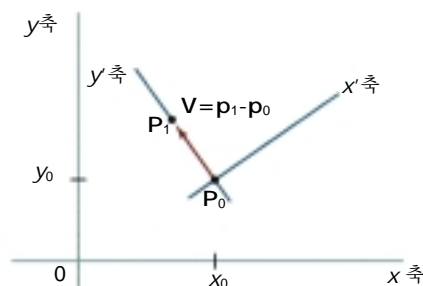



그림 5-29. xy 자리표계에서 두개의 자리표위치 P_0 및 P_1에 의하여 정의되는 $x'y'$직각자리표계

$$\mathbf{v} = \frac{\mathbf{P}_1 - \mathbf{P}_0}{|\mathbf{P}_1 - \mathbf{P}_0|} \quad (5-65)$$

와 같이 계산되며 u는 v에 수직이면서 오른손직각자리표계가 형성되게 얻어 진다.

6절. 아핀변환

형식

$$x' = a_{xx}x + a_{xy}y + b_x, \quad y' = a_{yx}x + a_{yy}y + b_y \quad (5-66)$$

의 자리표변환을 2차원아핀변환이라고 한다. 매개 변환된 자리표 x'와 y'는 본래 자리표 x와 y의 선형함수이며 파라메터 a_{ij}와 b_k는 변환의 형에 의하여 결정되는 상수이다. 아핀변환은 평행선은 평행선으로 변환되고 유한점은 유한점에로 넘어 간다는 일반적특징을 가진다.

평행이동, 회전, 비례변환, 반사 및 쏠림은 2차원아핀변환의 실례들이다. 임의의 일반적2차원아핀변환은 이 5개 변환들의 합성으로 항상 표현할수 있다. 다른 아핀변환은 한 자리표계로부터 다른 자리표계에로의 자리표서술의 변환이며 이것은 평행이동과 회전의 결합으로 서술할수 있다. 회전, 평행이동, 반사만을 포함하는 아핀변환은 평행선뿐아니라 각과 길이도 보존한다. 이 세개의 변환들에서는 길이와 두 선사이의 각이 변환후에 그대로 남는다.

7절. 변환함수

도형처리프로그람들은 매개 기본변환조작에 대하여 사용자들에게 절차 transformObject와 같은 개별적인 변환명령들을 제공하게 조직될수 있다. 이때 합성변환은 개별함수들을 변환순서의 요구대로 참조하면 설정된다. 또 다른 형식은 사용자들에게 매개 기본변환들에 대한 파라메터를 포함하는 하나의 변환함수를 제공하는것이다. 이 함수의 출력은 지적된 파라메터값들에 대한 합성변환행렬이다. 두개의 선택항목이 다 쓸모 있다. 개별함수는 간단한 변환조작에 편리하고 합성함수는 복잡한 변환순서를 지적하는데 편리한 방법이다.

PHIGS서고는 사용자에게 두가지 선택항목을 다 제공한다. 기본변환행렬을 발생시키는 개별명령들로서는

```
translate(translateVector, matrixTranslate)
rotate(theta, matrixRotate)
scale(scaleVector, matrixScale)
```

이다. 여기서 이 매 함수들은 자리표위치를 변환시키는데 리용될수 있는 동차렬벡토르로 표현되는 3×3변환행렬을 만든다. 파라메터 translateVector는 평행이동거리 t_x, t_y쌍에 대한 지시기이다. 류사하게 파라메터 scaleVector는 비례변환값 s_x ,s_y쌍을 지적한다. 회전 및 비례변환행렬(matrix Rotate와 matrix Scale)은 자리표원점에 대하여 변환된다.

앞에서 설정된 변환행렬들은 함수

```
composeMatrix(matrix2, matrix1, matrixOut)
```

에 의하여 결합된다. 여기서 합성된 출력행렬의 원소들은 matrix2를 matrix1에 왼쪽곱하기를 하는 방법으로 계산된다. 비례변환, 회전, 평행이동의 결합을 수행하는 합성변환행렬은 함수

```
buildTransformationMatrix (referencePoint, translateVector,
                           theta, scaleVector, matrix)
```

에 의하여 만들어 진다. 회전과 비례변환은 파라메터 referencePoint로 지적되는 자리표위치에 대하여 수행된다. 변환순서렬에 대한 순서는 (1) 비례변환, (2) 회전, (3) 평행이동으로 가정하며 합성변환의 원소들은 파라메터 matrix에 기억된다. 이 함수는 하나의 변환행렬 또는 둘 또는 세개 변환에 대한 합성행렬(언급된 순서로)을 얻는데 리용할수 있다. 평행이동행렬은 scale Vector=(1,1), theta=0 으로 설정하고 *x*, *y*이동값을 파라메터 translate Vector에 할당하면 얻을수 있다. 비례변환 또는 회전이 일어 나지 않을 때 파라메터 referencePoint는 변환계산에 영향을 미치지 않으므로 이 파라메터에는 임의의 자리표값을 설정할수 있다. 그러나 평행이동행렬만을 설정하려 한다면 함수 translate를 리용하고 간단히 평행이동벡토르를 지적할수 있다. 회전 및 비례변환행렬은 translateVector=(0,0)으로 설정하고 적당한 값들을 파라메터 referencePoint, theta, scaleVector에 할당하면 된다. 회전행렬만 얻기 위하여서는 scaleVector=(1,1)로 설정하며 비례변환만을 위하여서는 theta=0으로 설정한다. 자리표원점에 대하여 회전 또는 비례변환하려고 한다면 rotate 또는 scale함수를 리용하여 행렬을 설정하는것이 더 간단하다.

함수 buildTransformationMatrix는 항상 (1)비례변환, (2)회전, (3)평행이동 순서로 변환이 일어 나기때문에 다음의 함수는 다른 순서를 지적하기 위하여 제공된다.

```
ComposeTransformationMatrix (matrixIn, referencePoint,
             translateVector, theta, scaleVector, matrixOut)
```

이 함수는 buildTransformationMatrix함수 또는 임의의 변환순서를 합성하는 임의의 다른 행렬만들기함수와 결합하여 리용할수 있다. 실례로 고정점에 대한 비례변환행렬을buildTransformationMatrix함수에 의해 설정하고 다음에 이 비례변환행렬을 지적된 회전축점에 대한 회전과 결합하기 위하여 composeTransformationMatrix함수를 리용할수 있다. 합성된 회전비례변환순서는 이때 matrixOut에 기억된다.

변환행렬을 설정한후 행렬을 함수

```
transformPoint(inPoint, matrix, outPoint)
```

에 의해 물체의 개별적인 자리표위치에 적용할수 있다. 여기서 파라메터 inPoint에 물체점의 초기 *xy* 자리표위치를 주면 파라메터 outPoint에는 대응하는 변환된 자리표가 들어 간다. 2차원모형화변환을 수행하는데 사용할수 있는 추가적인 함수들은 제7장에서 론의한다.

8절. 변환에 대한 라스터방법

라스터체계의 고유한 능력들은 물체변환에 대한 또 다른 방법을 암시해 준다. 라스터체계는 그림정보를 프레임완충기에 화소무늬로 기억시킨다. 따라서 일부 간단한 변환들은 기억된 화소값들의 직4각형배렬을 프레임완충기안의 한 위치에서 다른데로 간단히 움직이는것에 의하여 빨리 수행할수 있다.

약간한 산수연산만 요구되며 따라서 화소변환이 아주 능률적이다.

직4각형화소배렬을 처리하는 라스터함수를 일반적으로 라스터옵스(raster ops)라고 한다. 화소들의 블로크를 한 위치에서 다른데로 움직이는것을 화소값들의 블로크평행이동이라고도 한다. 2값준위체계에서 특히 함수가 하드웨어적으로 실현될 때 이 조작을 bitBlt(비트-블로크평행이동 bit-block transfer의 략자)라고 한다. pixBlt라는 용어는 때때로 여러준위체계(화소당 여러비트)에서의 블로크평행이동에 대하여도 쓴다.

그림 5-30은 라스터구역의 블로크평행이동으로 수행되는 평행이동변환을 보여 주었다. 그림에 보여 준 직4각형구역안의 모든 비트설정이 라스터의 다른 부분에 블로크로 복사된다. 이 평행이동은 지적된 직4각형의 라스터구역에서 먼저 화소세기들을 읽어 배렬에 넣은 다음에 배렬을 새 라스터위치에 복사하는 방법으로 수행한다. 본래의 물체는 그의 직4각형구역을 배경세기로 채우면 지워 지게 된다(물체가 장면안의 다른 물체와 겹치지 않는다고 가정하고).

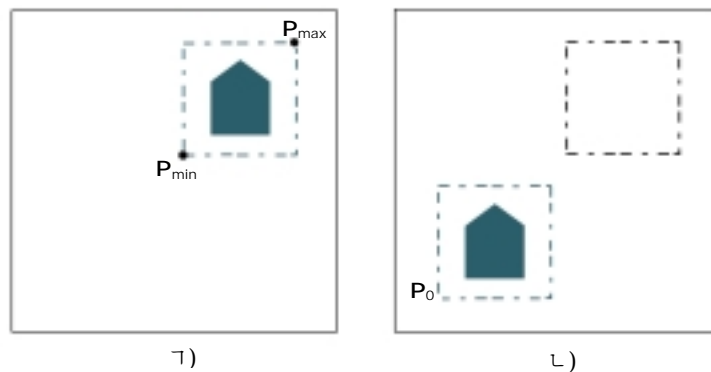



그림 5-30. 화소값들의 직 4 각형블로크의 움직임에 의한 화면위치 ㄱ로부터 위치 ㄴ에로의 물체평행이동(자리표위치 P_{min}과 P_{max}는 움직일 직 4 각형블로크의 한계를 지적하며 P_0은 목적기준위치이다.)

도형처리프로그람들에서 흔히 제공되는 대표적인 라스터함수들은 다음과 같다.

- copy(복사) - 화소블로크를 한 라스터구역으로부터 다른데로 움직인다.
- read(읽기) - 화소블로크를 지적된 배렬에 보관한다.
- write(쓰기) - 화소배렬을 프레임완충기의 어떤 위치에 써넣는다.

일부 실현들에서는 화소값결합에 대한 선택항목들을 준다. 교체방식에서 화소값은 목적위치에로 간단히 평행이동된다. 화소값들을 결합하는 다른 선택항목들로는 론리연산(and, or, exclusive or)과 2진산수연산이 있다. exclusive or 방식에서는 한 블로크를 동일한 라스터구역에 두번 련속하여 복사하면 그 구역에 있던 본래값이 재현된다. 이 기술은 배경을 지움이 없이 물체를 장면안에서 움직이는데 리용될수 있다. 화소값을 조정하는 다른 선택항목은 원천화소를 지적된 마스크와 결합시키는것이다. 이것은 블로크안의 선택된 위치들만 평행이동되게 하거나 또는 마스크에서 정의되는 무늬에 의하여 명암이 붙여 지게 한다.

90°회전은 블로크평행이동으로 쉽게 수행할수 있다. 먼저 배렬의 매행에서 화소들의 순서를 뒤집고 다음에 행과 렬을 서로 교체하면 물체를 시계바늘반대방향으로 90°회전시킬수 있다. 180°회전은 배렬의 매행에서 화소들의 순서를 뒤집고 다음에 행들의 순서를 뒤집으면 얻어 진다. 그림 5-31은 화소블로크를 90° 및 180°회전시키는데 필요한 배렬처리의 실례를 보여 준다.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \qquad \begin{bmatrix} 3 & 6 & 9 & 12 \\ 2 & 5 & 8 & 11 \\ 1 & 4 & 7 & 10 \end{bmatrix} \qquad \begin{bmatrix} 12 & 11 & 10 \\ 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

ㄱ) ㄴ) ㄷ)

그림 5-31. 화소값배렬의 회전(본래 배렬의 방향을 ㄱ에 보여 주고 시계바늘반대방향으로 90° 회전후 배렬의 방향을 ㄴ에 보여 주며 180° 회전후 배렬의 방향을 ㄷ에 보여 준다.)

90°의 배수가 아닌 배렬회전에 대하여서는 더 많은 계산을 하여야 한다. 일반적인 절차를 그림 5-32에서 설명한다. 매개 목적화소구역들은 회전되는 배렬에 넘겨 지며 회전되는 화소구역들과의 겹침량이 계산된다. 이때 목적화소의 세기는 겹치는 원천화소들의 세기를 구역겹침의 퍼센트에 의해 무게를 붙여 평균함으로써 계산한다.

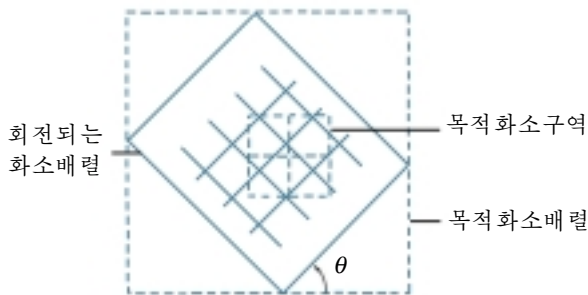



그림 5-32. 화소들의 직4각형블로크에 대한 라스터회전은 목적화소구역들을 회전되는 블로크에 넘기는 방법으로 진행한다.

화소블로크의 라스터비례변환은 3장 13절에서 설명한 세포배렬넘기기와 류사하다. 본래 블로크안의 화소구역을 지적된 s_x ,s_y값들을 리용하여 비례변환하고 비례변환된 직4각형을 목적화소들의 모임에 넘긴다. 매개 목적화소들의 세기는 다음에 그것의 비례변환된 화소구역과의 겹침구역에 따라 할당된다(그림 5-33).

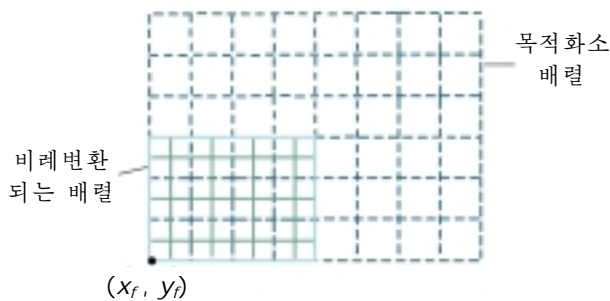



그림 5-33. 목적화소구역들을 비례변환되는 화소값들의 배렬에 넘긴다(비례결수 $s_x = s_y = 0.5$가 고정점 (x_f, y_f)에 대해 적용되고 있다).

요약

기본적인 기하학적변환들은 평행이동, 회전, 비례변환이다. 평행이동은 물체를 직선경로상에서 한 위치로부터 다른 곳으로 움직인다. 회전은 물체를 지적된 회전축점(회전점)주위로 원경로상에서 한 점으로부터 다른데로 평행이동시킨다. 비례변환은 지적된 고정점에 대한 물체의 치수를 변화시킨다.

2차원기하학적변환들은 변환들의 순서렬을 하나의 합성변환으로 결합시킬수 있는 3×3행렬연산으로 표현할수 있다. 이것은 물체의 합성행렬을 적용하여 초기자리표위치로부터 변환된 마지막위치를 얻는데서 계산량을 줄일수 있는 효과적인 형식화이다. 이를 위하여서는 2차원자리표위치들을 3요소렬벡토르나 행벡토르로 표현하여야 한다. 이 책에서는 자리표위치에 대한 렬벡토르표현을 선택한다. 그것은 이 표현이 표준수학표기이고 많은 도형처리프로그람들에서도 이 표기를 쓰기때문이다. 이때 2차원변환에서 자리표위치들은 세번째(동차)의 자리표에 1이 할당된 3요소동차자리표로 표현된다.

합성변환은 평행이동, 회전, 비례변환행렬들의 임의의 결합의 곱하기로 이루어 진다. 동화응용에서는 평행이동과 회전의 결합을 쓸수 있고 지적된 임의의 방향으로 물체들을 확대축소하는데는 회전과 비례변환의 결합을 사용할수 있다. 일반적으로 행렬곱하기에서는 교환칙이 성립되지 않는다. 실례로 평행이동회전순서렬의 순서를 변화시킨다면 결과가 달라 진다. 평행이동과 회전만을 포함하는 변환순서렬은 강체변환이다. 왜냐하면 각도들과 거리들이 변화되지 않기때문이다. 강체변환의 왼쪽웃부분행렬은 직교행렬이다. 그러므로 회전행렬들은 왼쪽 웃 2×2부분렬들을 2개의 직교하는 단위벡토르성분들로 설정하여 만들수 있다. 회전변환에서 회전각이 작을 때에는 시누스와 코시누스함수에 대한 근사계산을 써서 계산을 줄일수 있다. 그렇지만 회전걸음이 많아 지면 근사오차가 현저한 값으로 축적될수 있다.

기타 변환들로는 반사와 쏠림이 있다. 반사는 반사축에 대하여 물체를 180° 회전시키는 변환이다. 이것은 반사축에 대한 물체의 거울상을 낳는다. 반사축이 xy평면에 수직인 경우에는 반사가 xy평면에서의 회전처럼 얻어 진다. 반사축이 xy평면에 있을 때에는 반사가 xy평면에 수직인 평면상에서의 회전처럼 얻어 진다. 쏠림변환은 쏠림기준선으로부터의 자리표거리에 비례하는 량으로 x축 또는 y축 자리표값을 밀기하여 물체의 모양을 찌그러 지게 한다.

직각자리표계들사이의 변환은 평행이동-회전변환순서렬로 수행된다. 새로운 자리표계를 지적하기위한 한가지 방법은 새로운 자리표원점의 위치와 새로운 y축의 방향을 주는것이다. 이때 새로운 x축의 방향은 시계바늘방향으로 y방향벡토르를 90° 회전시켜 얻는다. 낡은 자리표계에서의 물체의 자리표서술은 새로운 자리표축을 낡은 자리표축에 일치시키는 변환행렬에 의해 변환된다. 이 변환행렬은 새로운 자리표원점을 낡은 자리표원점에 움직이는 평행이동과 두 자리표축들의 모임을 일치시키는 회전을 결합하여 계산할수 있다. 이 회전행렬은 새로운 자리표계의 x와 y방향단위벡토르로 얻어 지게 된다.

2 차원기하학적변환들은 아핀변환이다. 즉 그것들을 x와 y자리표들의 선형함수로 표현할수 있다. 아핀변환에서는 평행선은 평행선으로 변환되며 유한점들은 유한점들로 변환된다. 비례변환과 쏠림을 포함하지 않는 기하학적변환들도 역시 각들과 길이를 유지한다.

도형처리프로그람들에서는 보통 평행이동, 회전, 비례변환에 대해서만 변환함수들을 제공하고 있다. 이런 함수들에는 평행이동, 회전, 비례변환행렬들을 창조하는 개별적인 절차들과 변환순서렬에 대한 파라메터를 주면 합성행렬을 만드는 함수들이 들어 간다.

고속라스터변환들은 화소들의 블로크를 움직여서 수행할수 있다. 이것은 물체에 대한 변환자리표계산과 새로운 위치에 물체를 현시하기 위한 주사변환루틴을 적용하는것을 피하게 한다. 3가지 일반적인 라스터조작(bitBlts 와 pixBlts)은 복사, 읽기, 쓰기이다. 프레임완충기에서 화소들의 어떤 블로크를 새로운 위치에 옮기는 경우 단순히 낡은 화소값들을 교체시킬수도 있고 혹은 론리연산이나 산수연산을 적용하여 화소값들을 결합시킬수도 있다. 라스터평행이동은 프레임완충기에서 화소블로크를 새로운 위치에 복사함으로서 수행된다. 90°배수의 라스터회전은 블로크안에서 화소값들의 행과 렬위치를 조작하면 얻어 진다. 기타 회전들은 먼저 프레임완충기에서 회전되는 화소구역들을 목적위치에 넘기고 다음에 겹친구역들을 계산하면 된다. 라스터변환에서 비례변환도 변환되는 화소구역들을 프레임완충기의 목적위치에 넘기면 된다.

참고문헌

콤퓨터도형처리에서 동차자리표에 대한 보충적인 정보는 Blinn(1997과 1978)을 보시오.

PHIGS에서의 변환함수는 Hopgood과 Duce(1991), Howard(1991), Gaskins(1992), Blake(1993)에서 설명하였다. GKS변환함수들에 대한 정보는 Hopgood(1983) Enderle, Kansy, Pfaff(1984)을 보시오.

련습문제

5-1. 회전축점에 대하여 물체를 련속적으로 회전시키는 프로그람을 쓰시오. 매 련속된 회전에서는 작은 걸음각을 사용하며 시누스와 코시누스함수에 대해서는 계산속도를 높이기 위하여 근사식을 써야 한다. 매 걸음각은 하나의 완전한 순환이 30s이내에 수행되도록 선택하여야 한다. 자리표오차가 축적되는것을 피하기 위하여 새로운 순환이 시작될 때마다 물체의 초기자리표값은 재설정되게 하시오.

5-2. 두 회전의 합성은 $\mathbf{R}(\theta_1)$과 $\mathbf{R}(\theta_2)$의 행렬표현을 결합하면 다음과 같이 가법적이라는것을 보여 주시오.

$$\mathbf{R}(\theta_1)\cdot\mathbf{R}(\theta_2) = \mathbf{R}(\theta_1+\theta_2)$$

5-3. 임의로 입력되는 변환파라메터들의 모임에 대하여 합성변환행렬을 만드는 buildTransformationMatrix와 ComposeTransformationMatrix함수를 실현하기 위한 절차들을 쓰시오.

5-4. 현시되는 물체에 임의의 변환순서렬을 적용하는 프로그람을 짜시오. 사용자가 변환순서렬을 지적하고 관계되는 파라메터들을 차림표에서 선택하면 프로그람에서 합성변환이 계산되여 주어 진 물체를 변환하는데 리용되게 하시오. 본래의 물체와 변환된 물체를 서로 다른 색갈이나 무늬로 채우시오.

5-5. 임의의 방향으로 비례변환하는 식 5-35의 변환행렬을 수정하여 임의의 비례변환고정점 (x_f, y_f) 대한 자리표가 포함되도록 하시오.

5-6. 다음과 같은 연산들의 순서렬에서는 변환행렬의 곱하기가 교환적이라는것을 증명하시오.

ㄱ) 련속된 두번의 회전

ㄴ) 련속된 두번의 평행이동

ㄷ) 련속된 두번의 비례변환

5-7. 동형비례변환$(s_x=s_y)$과 회전은 교환할수 있는 조작쌍이지만 일반적으로 비례변환과 회전은 교환할수 없다는것을 증명하시오.

5-8. 비례변환, 회전, 평행이동행렬들을 개별적으로 곱하여 식 5-38의 합성변환행렬의 원소들을 얻으시오.

5-9. 식 5-51의 직선 $y=x$에 대한 반사변환행렬은 x축에 대하여 반사시킨 다음 시계바늘반대방향으로 90° 회전시킨것과 같다는것을 보여 주시오.

5-10. 식 5-52의 직선 $y=-x$에 대한 반사변환행렬은 y축에 대하여 반사시킨 다음 시계바늘반대방향으로 90° 회전시킨것과 같다는것을 보여 주시오.

5-11. 두 자리표축들에 대한 두번의 련속적인 반사는 자리표원점에 대한 한번의 회전과 같다는것을 보여 주시오.

5-12. 임의의 직선 $y = mx + b$에 대한 반사변환행렬의 형식을 규정하시오.

5-13. 자리표원점을 지나는 임의의 직선들에 대한 두번의 련속된 반사는 원점에 대한 한번의 회전과 같다는것을 보여 주시오.

5-14. 식5-53의 x방향쏠림변환행렬과 등가인 기본변환들의 순서렬을 결정하시오.

5-15. 식5-57의 y방향쏠림변환행렬과 등가인 기본변환들의 순서렬을 결정하시오.

5-16. 벡토르폰트로 정의된 주어 진 비낌체문자를 현시하기 위한 쏠림변환절차를 설정하시오. 즉 이 폰트에서 모든 문자들의 모양은 선분들로 정의되며 비낌체문자들은 쏠림변환에 의하여 만들어 진다. 몇개의 가능한 폰트들에서 비낌체본문과 일반본문을 비교하여 쏠림파라메터를 결정하시오.

5-17. 한 직각자리표계의 자리표점 $\mathbf{P}=(x, y)$를 그림 5-27에 보여 준것처럼 각 θ만큼 회전시켜 다른 직각자리표계의 자리표값 (x', y')로 변환하는 다음의 식을 유도하시오.

$$x' = x\cos\theta + y\sin\theta \qquad y' = -x\sin\theta + y\cos\theta$$

점 $\mathbf{P}$를 4개의 자리표축들에 투영하고 얻어 진 직3각형들을 해석하시오.

5-18. 물체의 서술을 한 직각자리표계로부터 다른 직각자리표계로 변환하는 행렬의 원소들을 계산하기 위한 절차를 쓰시오. 두번째 자리표계는 원점 $\mathbf{P}_0$과 이 체계의 정의 y'축방향벡토르 $\mathbf{V}$로 정의되여야 한다.

5-19. 프레임완충기의 직 4 각형구역의 블로크이동을 실현하기 위한 절차를 만드시오. 하나의 함수는 배렬에 읽어 들이는데 사용하고 다른 함수는 배렬을 지적된 이동구역에 복사하는데 사용하시오.

5-20. 여러가지 론리연산을 써서 프레임완충기의 같은 구역에 두번 련속블로크이동시킨 결과를 결정하시오.

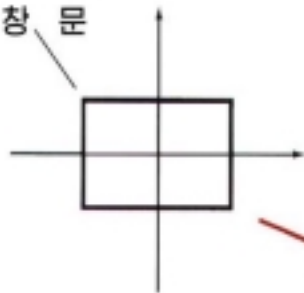
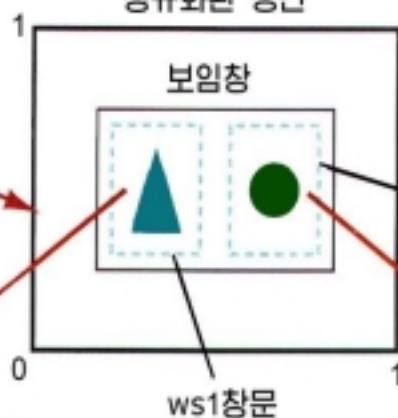
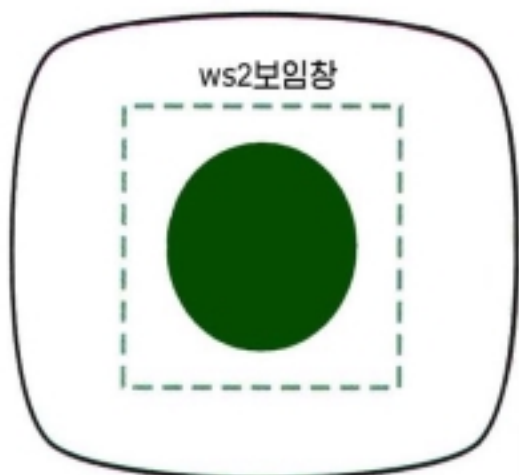
5-21. 2진산수연산을 써서 프레임완충기의 같은 구역에 두번 련속블로크이동시키면 결과가 어떻게 되는가를 보시오.

5-22. 임의로 지적되는 론리연산이나 치환(복사)연산을 써서 프레임완충기의 블로크평행이동을 수행하는 루틴을 실현하시오

5-23. 프레임완충기의 블로크이동에서 90°회전을 실현하는 루틴을 쓰시오.

5-24. 프레임완충기의 블로크이동에서 임의의 지적된 각도로 회전시키는 루틴을 쓰시오.

5-25. 화소블로크의 라스터변환으로 비례변환을 실현하는 루틴을 쓰시오.

6장. 2차원보기

보기자리표
창 문
정규화된 공간
1
보임창
ws2창문
0
1
ws1창문
ws1보임창
ws2보임창
현시장치 1
현시장치 2

이 장에서는 출력장치에 그림의 보임상을 현시하는 정규화된 수법을 고찰한다. 일반적으로 도형처리프로그람들은 사용자가 이미 정의된 그림의 어느 부분을 현시하며 현시장치에서 그 부분을 어디에 놓겠는가를 지적할수 있게 되여 있다. 세계자리표계라고 하는 임의의 편리한 직각자리표계가 그림을 정의하는데 리용될수 있다. 2차원그림에서 보임상은 전체 그림구역에서 부분구역을 지적하여 선택한다. 사용자는 현시를 위하여 하나의 구역을 선택할수도 있고 동시적인 현시나 동화에서 장면을 따르는 카메라의 움직임을 위하여 여러개 구역들을 선택할수도 있다. 선택된 구역안의 그림부분은 다음에 지적된 장치자리표구역에로 넘겨 진다. 다중보기구역이 선택되는 경우 이 구역들은 개별적으로 떨어 진 현시위치에 놓일수도 있고 일부 구역들이 다른 더 큰 현시구역들에 삽입될수도 있다. 세계자리표로부터 장치자리표에로의 변환은 선택된 현시구역의 범위밖에 있는 그림의 부분들을 지우는 절차뿐아니라 이동, 회전, 비례변환조작들도 포함한다.

1절. 보기과정의 흐름

현시를 위하여 선택되는 세계자리표구역을 **창문**(window)이라고 한다. 창문이 사영되는 현시장치의 구역을 **보임**창(viewport)이라고 한다. 창문은 무엇이 보여 지는가를 정의하며 보임창은 그것이 어디에 현시되는가를 정의한다. 흔히 창문과 보임창은 자리표축들에 평행인 경계를 가지는 표준위치의 직4각형들이다. 일반다각형형태와 원과 같은 다른 창문 또는 보임창형태들도 일부 응용들에서 리용되지만 이런 형태들은 처리가 오래다. 일반적으로 세계자리표의 장면의 일부를 장치자리표로 넘기는것을 **보기변환**이라고 한다. 때때로 2차원보기변환을 간단히 창문-보임창변환 또는 창문변환이라고도 한다. 그러나 일반적으로 보기는 창문으로부터 보임창에로의 변환보다 더 많은것을 포함하고 있다. 그림 6-1에서는 직4각형창문안에 떨어 지는 그림부분을 지적된 직4각형보임창에로 넘긴것을 보여 주었다.

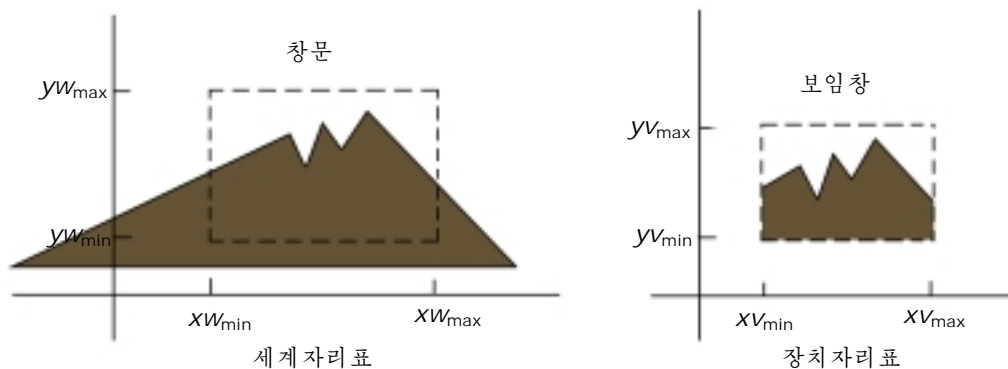



그림 6-1. 창문과 보임창에 표준직4각형을 리용하는 보기변환

콤퓨터도형처리술어로 창문은 본래 이 절의 처음에 정의한바와 같이 보기를 위하여 선택되는 그림의 구역을 가리킨다. 그러나 유감스럽게도 같은 용어가 지금 창문관리체계들에서는 주위로 움직일수 있고 크기를 다시 설정할수도 있으며 능동 또는 비능동으로도 할수 있는 임의의 직4각형화면구역을 가리키는데 쓰이고 있다. 이 장에서는 현시를 위하여 선택된 세계자리표장면의 구역을 가리키는 용어로서의 창문만을 리용한다. 화면창문과 창문관리체계에 대하여서는 8장에서 도형사용자대면부를 고찰할 때 설명한다.

창문 및 보임창조작을 제공해 주는 일부 도형처리프로그람들은 표준위치직4각형만을 허용하고

있는데 직4각형창문이 임의의 방향을 가지게 하는것이 보다 일반적이다. 이 경우 보기변환은 그림

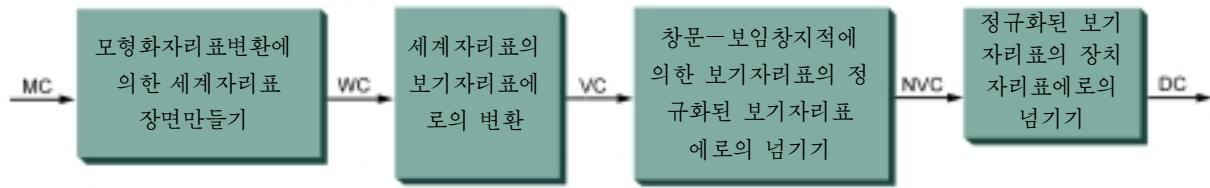



그림 6-2. 2 차원보기변환과정의 흐름

6-2에서 보는바와 같이 여러 걸음으로 수행된다. 먼저 3장과 4장에서 설명한 출력기초요소들과 속성들을 리용하여 세계자리표로 장면을 만든다. 다음 창문에 대한 개별적인 방향을 얻기 위하여 세계자리표평면에서 **2차원보기자리표계**를 설정하고 보기자리표계에서 창문을 정의한다. 보기자리표계는 직4각형창문에 대하여 임의의 방향을 설정하는 방법을 제공하는데 리용된다. 보기자리표계가 수립되면 세계자리표서술을 보기자리표에로 변환할수 있다. 다음에 정규화된 자리표(0부터 1사이 범위에서)로 보임창을 정의하고 장면의 보기자리표서술을 정규화된 자리표로 넘긴다. 마지막걸음에서 보임창밖에 있는 모든 그림부분들을 잘라 버리고 보임창의 내용을 장치자리표로 넘긴다. 그림 6-3에서는 회전된 보기자리표계와 정규화된 자리표에로의 넘기기를 보여 주었다.

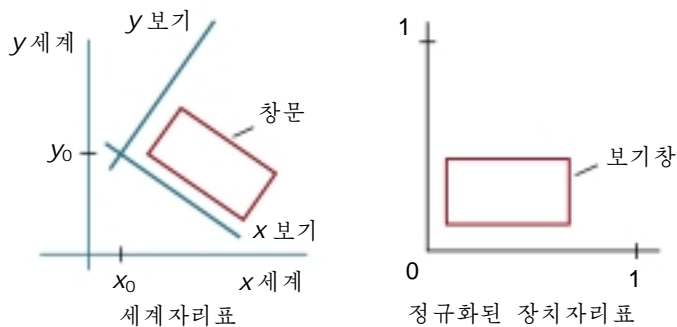



그림 6-3. 보기자리표에서 회전된 세계자리표창문의 설정과 대응하는 정규화된 자리표의 보임창

보임창의 위치를 변화시키면 출력장치의 서로 다른 위치의 현시구역에서 물체를 볼수 있다. 또한 보임창의 크기를 변화시킴으로써 현시되는 물체의 크기와 비례를 변화시킬수 있다. 서로 다른 크기의 창문들을 고정크기의 보임창에 련속적으로 넘기면 확대축소보기효과가 얻어 진다. 창문을 보다 작게 하여 큰 창문에서는 보이지 않았던 세부도 볼수 있게 장면의 일부분에 들어 가면 확대되여 보인다. 류사하게 축소해보기는 련속적으로 더 큰 창문에 의하여 장면의 부분으로부터 나오면 얻어 진다. 상하좌우 이동시켜 보는 효과는 고정크기의 창문을 장면의 여러 물체들을 가로질러서 움직이면 얻어 진다.

보임창은 일반적으로 단위크기의 바른4각형(정규화된 자리표)안에서 정의된다. 이것은 보기 그리고 기타 변환들이 개별적인 출력장치에 관계되지 않게 한다. 따라서 도형처리프로그람들은 대부분 장치와 독립적이다. 일단 장면이 정규화된 자리표로 변환되면 단위바른4각형은 그때에 리용하는 개별적인 출력장치의 현시구역에 간단히 넘겨 지게 된다. 해당한 장치구동기를 제공하여 주면 서로 다른 출력장치들을 리용할수 있다.

모든 자리표변환이 수행되면 보임창자르기를 정규화된 자리표나 장치자리표에서 수행할수 있다. 이것은 여러가지 변환행렬들을 결합할수 있게 하므로 계산을 줄인다. 자르기절차는 콤퓨터도형처리에서 아주 중요하다. 그것은 보기변환뿐아니라 창문관리체계와 그림그리기 및 작도프로그람들에서 지적된 화면구역의 내부 또는 외부의 그림부분을 지우는데와 기타 많은 응용들에서 리용된다.

2절. 보기자리표계

이 자리표계는 세계자리표창문을 지적하는 자리표계이다. 보기자리표계는 5장 5절에서 설명한 절차를 리용하여 설정한다. 먼저 보기자리표원점을 세계자리표의 어떤 위치 $\mathbf{P}_0=(x_0, y_0)$에 선택한다. 그다음 이 자리표계의 방향과 회전을 설정한다. 이렇게 하는 한가지 방법은 보기의 y_v방향을 정의하는 세계자리표벡토르 **V**를 지적하는것이다. 벡토르 **V**를 **보기벡토르**라고 한다.

V가 주어 지면 보기의 y_v 및 x_v축에 대하여 각각 단위벡토르 $\mathbf{v}=(v_x, v_y)$와 $\mathbf{u}=(u_x, u_y)$의 성분들을 계산할수 있다. 이 단위벡토르들은 x_vy_v보기자리표축을 x_wy_w세계자리표축과 일치시키는 회전행렬 **R**의 첫번째와 두번째 행을 형성하는데 리용한다.

세계자리표위치를 보기자리표로 변환하는 행렬은 두 걸음의 합성변환으로 얻는다. 먼저 보기자리표원점을 세계자리표원점에로 이동시키고 다음에 두 자리표계를 일치시키기 위하여 회전시킨다. 세계자리표를 보기자리표로 변환하는 2차원합성변환은

$$\mathbf{M}_{WC,VC} = \mathbf{R}\cdot\mathbf{T} \tag{6-1}$$

이다. 여기서 **T**는 보기자리표원점 $\mathbf{P}_0$을 세계자리표원점에로 가져 가는 이동행렬이고 **R**는 두 자리표계의 축들을 일치시키는 회전행렬이다. 그림 6-4에서는 이 자리표변환의 걸음들을 보여 주었다.

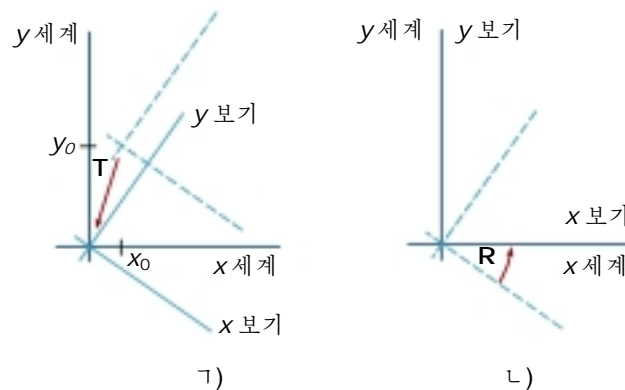



그림 6-4. 보기자리표계는 세계자리표계와 일치하도록 두 걸음으로 움직여 진다. ㄱ-보기자리표원점을 세계자리표원점에로 이동시킨다, ㄴ-두 자리표계의 축들이 일치하도록 회전시킨다.

3절. 창문-보임창자리표변환

물체서술이 일단 보기자리표계로 넘어 가면 창문의 범위를 보기자리표로 선택하며 보임창한계는 정규화된 자리표로 선택한다(그림 6-3). 그다음 물체서술을 정규화된 장치자리표로 변환한다. 이것은 정규화된 공간에서의 물체들의 상대적배치가 보기자리표에서와 같게 유지하는 변환을 리용하여 진행한다. 실례로 자리표위치가 보기창문의 중심에 있으면 그것은 보임창의 중심에 현시되게 한다.

그림 6-5에서는 창문-보임창넘기기를 보여 주었다. 창문에서 위치 (xw, yw)의 점은 관련된 보임창에서 위치 (xv, yv)에 넘어 간다. 보임창과 창문에서 같은 상대적배치를 유지하기 위하여서는

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$
$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}} \qquad (6-2)$$

를 요구한다. 이 식들을 보임창위치 (xv, yv)에 대하여 풀면

$$xv = xv_{min} + (xw - xw_{min})sx$$
$$yv = yv_{min} + (yw - yw_{min})sy \qquad (6-3)$$

를 얻는다. 여기서 비례결수는

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$
$$sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}} \qquad (6-4)$$

이다. 식 6-3은 또한 창문구역을 보임창구역에로 변환하는 변환들의 모임에 의해서도 유도할수 있다. 이 변환은 다음 순서의 변환들로 수행된다.

1. 고정점위치 (xw_{min}, yw_{min})에 대한 비례변환을 수행하여 창문구역을 보임창의 크기로 비례변환한다.
2. 비례변환된 창문구역을 보임창의 위치에로 이동시킨다.

그림 6-5. 지적된 창문에서 위치 (xw, yw)의 점은 보임창자리표 (xv, yv)에로 두 구역에서의 상대적위치가 갈도록 넘겨 진다.

물체의 상대적인 비례는 비례결수들이 같으면($sx = sy$) 유지된다. 그렇지 않으면 세계자리표물체는 출력장치에 현시될 때 x 또는 y방향으로 잡아 당겨 지거나 수축되게 된다.

문자렬은 보임창에 넘길 때 두가지 방법으로 조종할수 있다. 제일 간단한 넘기기는 설사 보임창구역이 창문에 비해 커지거나 또는 작아 져도 일정한 문자크기를 유지하는것이다. 이 방법은 본문이 변화될수 없는 표준문자폰트로 만들어 질 때 쓸수 있다. 문자크기를 변화시킬수 있는 체계들에서는 문자렬정의를 다른 기초요소들과 같이 창문화할수 있다. 선분으로 형성되는 문자들에 대한 보임창에로의 넘기기는 선변환순서렬과 같이 수행할수 있다.

물체서술은 정규화된 자리표로부터 여러가지 현시장치에로 넘겨 지게 된다. 개별적인 응용에서는 임의의 개수의 출력장치들이 열릴수 있으며 매개 열린 출력장치에 대하여 서로 다른 창문-보임창변환이 수행될수 있다. **워크스테이션변환**이라고 하는 이 넘기기는 정규화된 공간에서 창문구역을 선택하고 현시장치의 자리표로 보임창구역을 선택함으로써 수행된다. 워크스테이션변환에 의하여 장면의 부분들이 개별적인 출력장치들에서 위치를 정할 때 일부 보충적인 조종을 받게 된다. 그림 6-6에서

설명되는바와 같이 워크스테이션변환은 정규화된 공간의 서로 다른 부분들이 서로 다른 출력장치에 현시될수 있게 하는데 리용할수 있다.

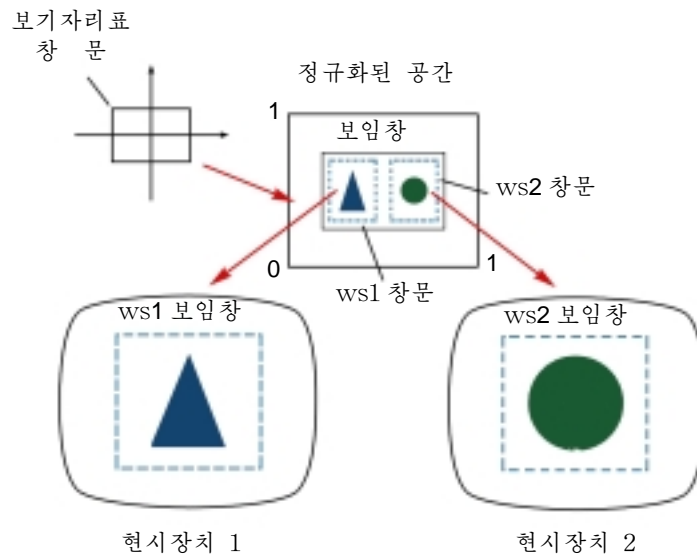



그림 6-6. 워크스테이션변환에 의하여 정규화된 자리표에서 선택된 장면의 부분들의 서로 다른 영상현시장치에로의 넘기기

4절. 2차원보기함수

PHIGS응용프로그람에서는 보기자리표계를 다음의 함수에 의하여 정의한다.

```
evaluateViewOrientationMatrix (x0, y0, xv, yv,
                               error, viewMatrix)
```

여기서 파라메터 x0과 y0은 보기자리표원점의 자리표이며 파라메터 xv, yv는 보기벡토르의 세계자리표위치이다. 옹근수오유코드는 입력파라메터들이 오유일 때 발생되며 그렇지 않으면 세계-보기변환에 대한 viewMatrix가 계산된다. 하나의 응용에서 임의의 개수의 보기변환행렬이 정의될수 있다.

창문-보임창넘기기행렬의 원소들을 설정하기 위하여서는 함수

```
evaluateViewMappingMatrix (xwmin, xwmax, ywmin, ywmax,
    xvmin, xvmax, yvmin, yvmax,error, viewMappingMatrix)
```

를 리용한다. 여기서 보기자리표로 창문의 한계가 파라메터 xwwmin, xwmax, ywmin, ywmax에 의하여 선택되며 보임창의 한계는 정규화된 자리표위치 xvmin, xvmax, yvmin, yvmax에 의하여 설정된다. 보기변환행렬에서와 마찬가지로 여러개의 창문-보임창쌍들을 만들수 있는데 그것들은 장면의 여러 부분들을 단위바른4각형의 서로 다른 구역에 투영하는데 리용할수 있다.

다음으로 여러 워크스테이션에 대한 보기 및 창문-보임창넘기기결합은

```
setViewRepresentation (ws, viewIndex, viewMatrix,
```

```
      viewMappingMatrix, xclipmin, xclipmax, yclipmin,
            yclipmax, clipxy)
```

에 의하여 보기표에 기억시킬수 있다. 여기서 파라메터 ws는 출력장치(워크스테이션)를 가리키며 파라메터 viewIndex는 이 개개 창문-보임창쌍에 대하여 옹근수식별자를 설정한다. 행렬 viewMatrix와 viewMappingMatrix는 viewIndex에 의하여 결합 및 참조될수 있다. 이 함수에서 보충적으로 자르기한계도 지적되여 있는데 그것들은 보통 보임창경계와 일치되도록 설정한다. 그리고 파라메터 clipxy에는 값 noclip(자르지 않음) 또는 clip(자름)가 할당된다. 이것은 보임창밖의 장면부분을 보려는 경우 자르기를 하지 않게 한다. 또한 모든 장면이 보임창경계안에 들어 간다는것을 알 때에는 처리속도를 높이기 위하여 noclip를 선택할수 있다.

함수

```
setViewIndex (viewIndex)
```

는 보기표로부터 선택항목들의 개별적인 모임을 선택한다. 이 보기첨수viewIndex선택은 후에 지적되는 출력기초요소들과 련관속성들에 적용되여 매개 능동워크스테이션에서 화면을 발생시킨다.

마지막으로 워크스테이션창문보임창쌍을 선택하여 워크스테이션변환을 적용한다.

```
SetWorkstationWindow (ws, xwsWindmin, xwsWindmax,
                          ywsWindmin, ywsWindmax)
setWorkstationViewport (ws, xwsVPortmin, xwsVPortmax,
                          ywsVPortmin, ywsVPortmax)
```

여기서 파라메터 ws는 워크스테이션번호를 준다. 창문자리표범위는 0～1사이(정규화된 공간)에서 지적되며 보임창한계는 옹근수장치자리표로 지적된다.

워크스테이션보임창이 지적되지 않으면 정규자리표계의 단위바른4각형은 출력장치에서 가능한 제일 큰 바른4각형구역에로 넘어 간다. 정규화된 공간의 자리표원점은 장치자리표원점에 넘겨 지며 단위바른4각형은 출력장치의 바른4각형구역에 변환되므로 종횡비가 보존된다.

실례 6-1. 2 차원보기실례

다음의 명령순서렬은 보기함수들의 사용실례로서 세계자리표에서 회전된 창문을 설정하고 그 내용을 워크스테이션 2의 오른쪽 웃구석에 넘긴다. 보기자리표원점은 세계자리표원점으로 잡고 창문에 대한 보기방향은 (1,1)로 선택한다. 이것은 세계자리표에서 시계바늘방향으로 45˚ 회전된 보기자리표계를 준다. 보기첨수값은 5로 선택한다.

```
evaluateViewOrientationMatrix (0, 0, 1, 1,
                                  viewError, viewMat);
evaluateViewMappingMatrix (-60.5, 41.24, -20.75, 82.5, 0.5,
           0.8, 0.7, 1.0, viewMapError, viewMapMat);
setViewRepresentation (2, 5, viewMat, viewMapMat, 0.5, 0.8,
                                     0.7, 1.0, clip);
```

```
setViewIndex (5);
```

류사하게 지적된 창문을 화면의 왼쪽 아래구석의 보임창에 넘기는 보기첨수 6을 가지는 추가적인 변환이 설정된다. 그다음에 실례로 2개의 그라프를 반대쪽 화면구석에 현시하게 된다.

```
setViewIndex (5)
polyline (3,axes);
polyline (15,datal);
setViewIndex (6)
polyline (3,axes);
polyline (25,data2);
```

보기첨수 5는 화면의 오른쪽 웃구석에 보임창을 선택하고 보기첨수 6은 왼쪽 구석에 보임창을 선택한다. 함수 polyline (3, axes)은 매 그라프에서 자료를 현시하기 위한 수직자리표축을 만든다.

5절. 자르기조작

일반적으로 그림의 부분이 지적된 공간구역의 내부인가 또는 외부인가를 식별하는 모든 절차를 **자르기알고리듬**(cliping algorithm) 또는 간단히 **자르기**라고 한다. 물체가 잘라 지게 될 구역을 **자르기창문**이라고 한다.

자르기는 보기를 위하여 정의된 장면의 일부분 뽑아 내기, 3차원보기에서 보이는 면의 식별, 선분 또는 물체경계의 거치름제거, 고체모형화절차를 리용한 물체생성, 다중창문환경현시, 그림부분의 복사, 이동, 지우기, 2중화를 선택할수 있는 작도 및 그림그리기조작 등에 응용된다. 응용에 따라 자르기창문은 일반다각형일수도 있고 지어 곡선경계를 가질수도 있다. 먼저 직4각형자르기구역을 리용하는 자르기방법을 고찰하고 다음에 다른 자르기구역형태에 의한 방법들을 설명한다.

보기변환에 의하여 창문구역안에 있는 그림부분들만 현시하려 한다면(자르기기발이 noclip로 설정되지 않았다고 가정하면) 창문밖의 모든것은 버리게 된다. 자르기알고리듬을 세계자리표에서 적용하고 다음에 창문내부의 내용들만 장치자리표에 넘길수 있다. 또 다르게 완전한 세계자리표그림을 먼저 장치자리표 또는 정규화된 장치자리표에 넘기고 다음에 보임창경계에 의하여 잘라 버릴수도 있다. 세계자리표자르기를 하면 창문밖의 기초요소들은 앞으로 고찰하지 않아도 되며 따라서 그 기초요소들을 장치공간에로 변환하는 처리가 없어 진다. 한편 보임창자르기는 보기와 기하변환행렬들을 결합할수 있으므로 계산을 줄일수 있다. 그러나 보임창자르기에서는 창문구역밖에 있는것들까지 포함하여 모든 물체에 대하여 장치자리표변환을 해야 한다. 라스터체계에서 자르기알고리듬은 흔히 주사변환과 결합된다.

다음 절들에서는 다음의 기초요소형들의 자르기에 대한 알고리듬을 고찰한다.

- 점자르기
- 선자르기(선분)
- 구역자르기(다각형)
- 곡선자르기
- 본문자르기

선 및 다각형자르기루틴은 도형처리프로그람들의 표준구성요소이다. 그러나 많은 프로그람들은 곡선물체 특히 스플라인곡선과 원 및 타원과 같은 원추곡선도 처리할수 있다. 곡선물체를 다루는 다른 방법은 그것들을 선분들로 근사화하고 선 또는 다각형자르기절차를 적용하는것이다.

6절. 점자르기

자르기창문이 표준위치의 직4각형이라고 가정하면 다음의 부등식들이 만족될 때 점 **P** =(x, y)는 현시해야 할 점으로서 보관한다.

$$\begin{aligned} xw_{\min} \le x \le xw_{\max} \\ yw_{\min} \le y \le yw_{\max} \end{aligned} \qquad (6-5)$$

여기서 자르기창문의 경계 ($xw_{\min}$, $xw_{\max}$, $yw_{\min}$, $yw_{\max}$)는 세계자리표의 창문경계 또는 보임창경계일수 있다. 이 4개의 부등식중 어느 하나라도 만족되지 않으면 점은 잘라 버린다(현시하지 말아야 할 점으로서 보관하지 않는다.).

일부 응용들에서 점자르기절차를 쓰지만 일반적으로 점자르기는 선 또는 다각형자르기보다 적게 리용된다. 실례로 점자르기는 장면의 일부 구역에 분포되는 립자(점)들로 모형화되는 폭발 또는 바다물거품을 포함하는 장면들에 적용될수 있다.

7절. 선자르기

그림 6-7은 직선과 표준직4각형자르기구역사이에서 있을수 있는 관계를 설명한다. 선자르기절차에는 여러가지 부분들이 포함된다. 먼저 주어 진 선분이 자르기창문안에 완전히 놓이는가를 검사할수 있다. 만약 그렇지 않으면 창문밖에 완전히 놓이는가를 검사한다. 마지막으로 선이 완전히 내부 또는 완전히 외부에 놓인다는것을 식별할수 없다면 하나 또는 그이상의 자르기경계와의 사귐점계산을 하여야 한다. 선의 끝점들의《내부외부》검사를 통하여 선을 처리한다. $\mathbf{P}_1$부터 $\mathbf{P}_2$까지의 선과 같이 두 끌

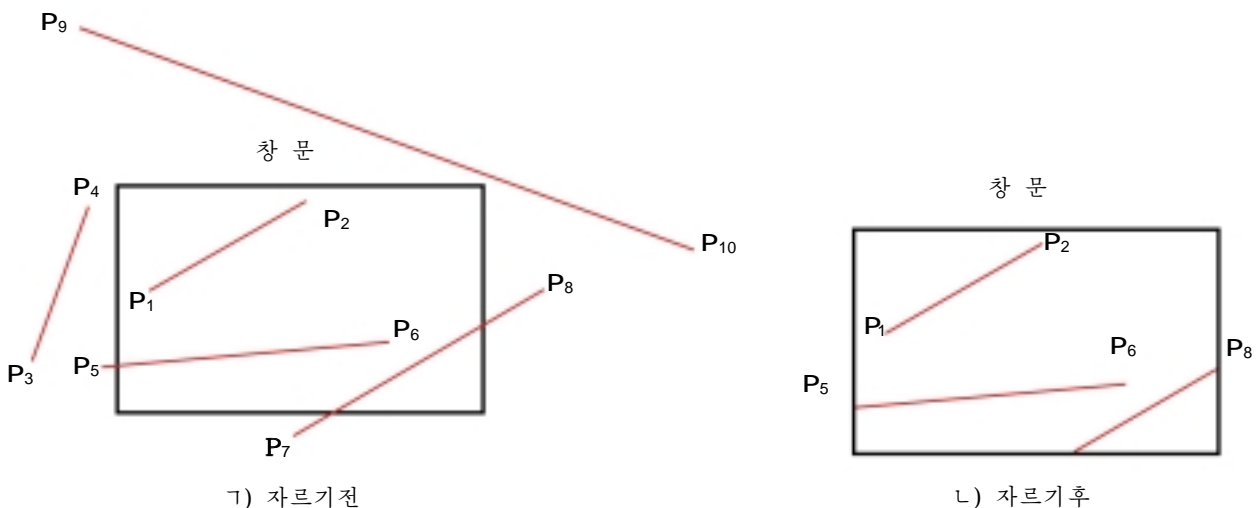



그림 6-7. 직4각형자르기창문에 대한 선자르기

점들이 다 모든 자르기경계의 내부에 있는 선은 보관된다. 두 끝점들이 다 어느한 자르기경계밖에 있는 선(그림 6-7에서 선 $\overline{\mathbf{P}_3\mathbf{P}_4}$)은 창문밖에 있다. 하나 또는 그이상의 자르기경계를 가로지르는 다른 모든 선들은 다중사귐점계산을 요구할수 있다. 계산을 최소화하기 위하여 외부선을 효과적으로 식별하며 사귐점계산을 줄일수 있는 자르기알고리듬이 요구된다.

끝점들이 (x_1, y_1), (x_2, y_2)이고 하나 또는 두 끝점이 다 자르기직4각형밖에 있는 선분인 경우 보조변수방정식

$$x = x_1 + u(x_2 - x_1)$$
$$y = y_1 + u(y_2 - y_1), \quad 0 \le u \le 1 \qquad (6-6)$$

을 리용하여 해당한 자르기경계와의 사귐점에 대한 보조변수 u의 값을 결정한다. 직4각형경계와의 사귐점에서 u의 값이 0～1범위밖에 있으면 선은 그 경계에서는 창문내부에 들어 가지 않는다. u의 값이 0～1범위안에 있으면 선분은 실제로 자르기구역을 통과한다. 선분의 어느 부분이 현시되는가를 결정하기 위하여 매개 자르기경계선에 대하여 이 방법을 차례로 적용하게 된다. 창문경계에 평행인 선분은 특별한 경우로 처리한다.

이 보조변수검사에 의한 선분의 자르기는 많은 량의 계산을 요구한다. 자르기를 보다 빨리 할수 있는 방법들도 있다. 많은 능률적인 선자르기방법들이 개발되였다. 다음 부분에서 중요한 알고리듬들을 개괄한다. 일부 알고리듬들은 명백히 2차원적그림에 대하여 설계된것들이며 일부는 3차원응용에 쉽게 적응되는것들이다.

코헨-싸더랜드선자르기

이것은 제일 먼저 나온것으로서 보편적인 선자르기절차들중의 하나이다. 일반적으로 이 방법은 계산해야 할 사귐점의 수를 줄일수 있는 내부검사를 진행함으로써 선분에 대한 처리속도를 높인다. 그림의 매개 선 끝점들에는 **구역코드**라고 하는 4자리2진코드가 할당되는데 그것은 자르기직4각형의 경계에 대한 점의 상대적위치를 식별한다. 구역들은 그림 6-8에 보여 준바와 같이 경계들을 기준으로 설정한다. 구역코드에서 매개 비트위치는 자르기창문에 대한 점의 4가지 상대적자리표위치 즉 왼쪽, 오른쪽, 우, 아래중 하나를 지적한다. 구역코드에서 비트위치들에 오른쪽에서부터 왼쪽으로 1부터 4까지의 번호를 매기면 자리표구역들이 비트위치와

bit 1 : 왼쪽	bit 2 : 오른쪽
bit 3 : 아래	bit 4 : 우

와 같이 호상관계를 가질수 있다. 임의의 비트위치에서 값 1은 그 점이 그 상대위치에 있다는것을 가리킨다. 그렇지 않으면 비트위치는 0으로 설정된다. 점이 자르기직4각형안에 있으면 구역코드는 0000이다. 직4각형의 왼쪽 아래에 있는 점은 구역코드 0101을 가진다.

그림 6-8. 자르기직4각형에 대한 상대위치에 따라 선의 끝점에 할당되는 2진구역코드

구역코드의 비트값은 끝점자리표값 (x, y)을 자르기경계와 비교하여 결정한다. 비트1은 $x < xw_{min}$이면 1로 설정한다. 다른 세개의 비트값들도 류사한 비교를 써서 결정할수 있다. 비트처리가 가능한 언어에서는 구역코드의 비트값들을 다음의 두 걸음에 의하여 결정할수 있다. (1)끝점자리표와 자르기경계사이의 차를 계산한다.

(2) 계산결과의 부호비트를 리용하여 구역코드에서 대응하는 값을 설정한다. 비트1은 $x-xw_{min}$의 부호비트, 비트2는 $xw_{max}-x$의 부호비트, 비트3은 $y-yw_{min}$의 부호비트, 비트4는 $yw_{max}-y$의 부호비트이다.

모든 선끝점들에 대하여 구역코드가 설정되면 어느 선이 자르기창문내부에 완전히 있으며 어느것이 명백히 바깥에 있는가를 빨리 결정할수 있다. 창문경계안에 완전히 포함되는 모든 선은 두 끝점이 다 구역코드 0000을 가진다. 이 선들은 무난하게 받아 들인다. 두 끝점의 구역코드들이 같은 비트위치에서 1을 가지는 선은 모두다 완전히 자르기직4각형밖에 놓인다. 따라서 이 선들은 버린다. 실례로 한 끝점의 구역코드가 1001이고 다른 끝점의 코드가 0101인 선은 버린다. 이 선의 두 끝점의 구역코드들은 첫번째 비트위치에서 다같이 1로 지적되여 있으며 자르기직4각형의 왼쪽에 있다. 총적으로 자르기를 위한 선검사에 리용할수 있는 방법은 두 구역코드에 대하여 론리and연산을 진행하는것이다. 결과가 0000이 아닌 선은 완전히 자르기구역밖에 있다.

이 검사에 의하여 자르기창문의 완전한 내부인가 또는 완전한 외부인가를 식별할수 없는 선들은 창문경계와의 사귐점을 위하여 검사한다. 그림 6-9에 보여 준바와 같이 이러한 선들은 창문내부를 가로지르거나 그렇지 않을수도 있다. 선에 대한 자르기처리는 선의 얼마만한 부분을 버려야 하는가를 결정하기 위하여 외부끝점을 어떤 자르기경계와 비교하는것으로부터 시작한다. 다음에 선의 남은 부분을 다른 경계들에 대하여 검사하는데 선이 총체적으로 버려 지거나 또는 일부분이 창문내부에서 발견될 때까지 계속한다. 알고리듬은 선끝점을 자르기경계에 대하여 왼쪽, 오른쪽, 아래, 우의 순서로 검사하도록 설정한다.

직4각형경계로 선을 자르는데서 코헨-싸더랜드알고리듬을 리용한 구체적인 걸음들을 설명하기 위하여 그림 6-9에서 선들이 어떻게 처리될수 있는가를 보자.

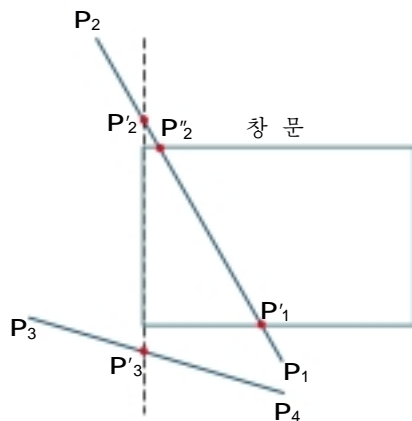



그림 6-9. 한 자르기구역에서 다른데로 나가는 선은 자르기창문을 통과하거나 또는 창문에 들어 감이 없이 자르기경계를 지나갈수 있다.

$\mathbf{P}_1$에서 $\mathbf{P}_2$까지의 선의 아래점에서 시작하여 $\mathbf{P}_1$를 왼쪽, 오른쪽, 아래 경계에 대하여 차례로 검사하여 이 점이 자르기직4각형의 아래에 있다는것을 알아 낸다. 다음에 아래경계와의 사귐점 $\mathbf{P}_1'$를 찾아내고 $\mathbf{P}_1$부터 $\mathbf{P}_1'$까지의 선부분을 버린다. 이제는 선이 $\mathbf{P}_1'$부터 $\mathbf{P}_2$까지의 부분으로 줄어 들었다. $\mathbf{P}_2$가 아직 자르기창문의 밖에 있기때문에 이 끝점을 경계들에 대하여 검사하여 그것이 창문의 왼쪽에 있다는것을 알아 낸다. 사귐점 $\mathbf{P}_2'$가 계산되지만 이 점은 창문의 우에 있다. 그러므로 마지막사귐점계산으로 $\mathbf{P}_2''$를 얻어 $\mathbf{P}_1'$부터 $\mathbf{P}_2''$까지의 선을 보관한다. 이것으로 이 선에 대한 처리가 끝난다. 이 부분을 보관하고 다음 선에로 간다. 다음 선에서 점 $\mathbf{P}_3$는 자르기직4각형의 왼쪽에 있으므로 사귐점 $\mathbf{P}_3'$를 결정하고 $\mathbf{P}_3$부터 $\mathbf{P}_3'$까지의 선부분을 없앤다. $\mathbf{P}_3'$부터 $\mathbf{P}_4$까지의 선부분에 대하여 구역코드를 검사하면 선의 나머지도 자르기창문의 아래에 있다는것을 찾아 내여 역시 버린다.

자르기경계와의 사귐점은 방향결수직선의 방정식을 리용하여 계산할수 있다. 끝점자리표가 (x_1, y_1), (x_2, y_2)인 선에 대하여 수직경계와의 사귐점의 y자리표는

$$y = y_1 + m(x - x_1) \tag{6-7}$$

의 계산으로 얻을수 있다. 여기서 x값은 xw_{min} 또는 xw_{max}로 설정되며 선의 경사도는 $m=(y_2-y_1)/(x_2-x_1)$로 계산된다. 류사하게 수평경계와의 사귐점의 x자리표는

$$x = x_1 + \frac{y - y_1}{m} \tag{6-8}$$

와 같이 계산할수 있다. 여기서 y는 yw_{min} 또는 yw_{max}로 설정된다.

다음의 프로그람은 코헨-싸더랜드선자르기알고리듬을 보여 준다. 매개 끝점에 대한 코드는 바이트로 기억되며 비트조작으로 처리된다.

```
#define ROUND(a) ((int)(a+0.5))

/* Bit masks encode a point’s position relative to the clip edges.
   A point’s status is encoded by OR'ing together appropriate
   bit masks. */

#define LEFT_EDGE 0x1
#define RIGHT_EDGE 0x2
#define BOTTOM_EDGE 0x4
#define TOP_EDGE 0x8

/* Points encoded as 0000 are completely Inside the clip rectangle;
   all others are outside at least one edge. If OR’ing two codes is
   FALSE (no bits are set in either code), the line can be Accepted.
   If the AND operation between two codes is TRUE, the line
   defined by those endpoints is completely outside the clip
   region and can be Rejected. */

#define INSIDE (a) (!a)
#define REJECT(a,b) (a&b)
#define ACCEPT(a,b) (!(a|b))

unsigned char encode (wcPt2 pt, dcPt winMin, dcPt winMax)
{
  unsigned char code=0x00;

  if (pt.x < winMin.x)
     code = code | LEFT_EDGE;
  if (pt.x > winMax.x)
     code = code | RIGHT_EDGE;
  if (pt.y < winMin.y)
     code = code | BOTTOM_EDGE;
  if (pt.y > winMax.y)
     code = code | TOP_EDGE;
  return (code) ;
```

```
}

void swapPts (wcPt2 * pi, wcPt2 * p2)
{
  wcPt2 tmp;

  tmp = *pl; *pl = *p2; *p2 = tmp;
}

void swapCodes (unsigned char * cl, unsigned char * c2)
{
  unsigned char tmp;

  tmp = *cl; *cl = *c2; *c2 = tmp;
}

void clipLine (dcPt winMin, dcPt winMax, wcPt2 pi, wcPt2 p2)
{
  unsigned char codel, code2;
  int done = FALSE, draw = FALSE;
  float m;

  while (!done) {
   codel = encode (pi, winMin, winMax);
     code2 = encode (p2, winMin, winMax);
     if (ACCEPT (codel, code2)) {
        done = TRUE ;
        draw = TRUE;
    }
    else
      if (REJECT (codel, code2))
        done = TRUE ;
      else {
        /* Ensure that pi is outside window */
        if (INSIDE (codel)) {
        swapPts (&pl, &p2) ;
        swapCodes (&:codel, &code2);
    }

    /* Use slope (m) to find line-clipEdge intersections */
    if (p2.x != pi.x)
       m = (p2.y - pl.y) / (p2.x - pl.x);
    if (codel & LEFT_EDGE) {
       pl.y += (winMin.x - pl.x) * m;
       pl.x = winMin.x;
    }
    else
```

```
        if (codel & RIGHT_EDGE) {
          pl.y += (winMax.x - pl.x) * m;
          pl.x = winMax.x;
        }
        else
          if (codel & BOTTOM_EDGE) {
            /* Need to update pl.x for non-vertical lines only */
             if (p2.x != pl.x)
               pl.x += (winMin.y - pl.y) / m;
             pl.y = winMin.y;
          }
          else
             if (codel & TOP_EDGE) {
               if (p2.x [= pi.x)
                  pl.x += (winMax.y - pl.y) / m;
               pl.y = winMax.y;
             }
      }
    }
    if (draw)
      lineDDA (ROUND(pi.x), ROUND(pl.y), ROUND(p2.x), ROUND(p2.y));
  }
```

리앙-바스키선자르기

선분의 보조변수방정식에 대한 분석에 기초한 보다 빠른 선자르기방법이 개발되였다. 그것을

$$x = x_1 + u\Delta x$$
$$y = y_1 + u\Delta y, \qquad 0 \le u \le 1 \tag{6-9}$$

의 형식으로 쓸수 있다. 여기서 $\Delta x = x_2 - x_1$, $\Delta y = y_2 - y_1$이다. 이 보조변수방정식을 리용하여 씨루스(Cyrus)와 벡크(Beck)는 코헨-싸더랜드알고리듬보다 일반적으로 더 효과적인 알고리듬을 개발하였다. 후에 리앙과 바스키가 독립적으로 더욱 더 빠른 보조변수선자르기알고리듬을 창안하였다. 리앙-바스키(Liang-Barsky)방법에 따라 먼저 점자르기조건식 6-5를 보조변수형식

$$xw_{\min} \le x_1 + u\Delta x \le xw_{\max}$$
$$yw_{\min} \le y_1 + u\Delta y \le yw_{\max} \tag{6-10}$$

으로 쓰자. 이 4개의 매 부등식은

$$up_k \le q_k, \qquad k = 1, 2, 3, 4 \tag{6-11}$$

와 같이 표현할수 있다. 여기서 파라메터 p와 q는

$$p_1 = -\Delta x, \quad q_1 = x_1 - xw_{\min}$$
$$p_2 = \Delta x, \quad q_2 = xw_{\max} - x_1$$
$$p_3 = -\Delta y, \quad q_3 = y_1 - yw_{\min} \tag{6-12}$$
$$p_4 = \Delta y, \quad q_4 = yw_{\max} - y_1$$

와 같이 정의된다. 자르기경계중 어느 하나의 평행인 선은 그 경계에 대응하는 k값에 대하여 $p_k=0$을 가진다($k=1,2,3,4$ 는 각각 왼쪽, 오른쪽, 아래, 웃경계에 대응한다). 그 k값에 대하여 또한 $q_k < 0$을 찾

아 내면 선은 완전히 경계밖에 있으므로 앞으로의 고찰에서 빼 버릴수 있다. $q_k \geq 0$이면 선은 두 평행 자르기경계안에 있다.

$p_k < 0$일 때 선의 무한연장은 그 자르기경계의 무한연장의 외부로부터 내부에로 향한다. $p_k > 0$이면 선은 내부로부터 외부에로 향한다. p_k의 령아닌값에 대하여 무한히 연장된 선이 경계 k의 연장선을 가로지르는 점에 대응하는 u의 값을

$$u = \frac{q_k}{p_k} \tag{6-13}$$

와 같이 계산할수 있다.

매 선에 대하여 자르기직4각형안에 있는 선의 부분을 정의하는 파라메터 u_1과 u_2의 값을 계산할수 있다. u_1의 값은 선이 외부로부터 내부로 들어 오는($p<0$) 직4각형경계를 찾아 결정한다. 이 경계들에 대하여 $r_k=q_k/p_k$를 계산한다. u_1의 값은 0과 r의 여러가지 값으로 이루어 지는 모임에서 제일 큰것으로 취한다. 반대로 u_2의 값은 선이 내부로부터 외부로 나가는($p>0$) 경계들을 시험하여 결정한다. r_k의 값은 이 매개 경계에 대하여 계산되며 u_2의 값은 1과 계산된 r값들로 이루어 지는 모임에서 제일 작은 값이다. $u_1>u_2$이면 선은 완전히 자르기창문밖에 있으며 버릴수 있다. 그렇지 않으면 잘라 진 선의 끝점들은 파라메터 u의 두 값으로부터 계산된다.

이 알고리듬을 다음의 프로그람에서 보여 준다. 선의 사귐점파라메터들은 값 $u_1=0$, $u_2=1$로 초기화된다. 매개 자르기경계에 대하여 p와 q에 대한 해당한 값들을 계산하고 선을 버릴수 있는가 또는 사귐점파라메터가 조정되는가를 결정하는 함수 clipTest에서 리용한다. 파라메터 r는 $p<0$이면 u_1을 고치는데 리용하고 $p>0$이면 u_2를 고치는데 리용한다. u_1 또는 u_2를 고친 결과 $u_1>u_2$이면 그 선은 버린다. 그렇지 않으면 새 값이 선을 짧게 하는 결과를 가져올 때에만 해당한 u파라메터를 고친다. $p=0$이고 $q<0$이면 그 선은 이 경계에 평행이고 밖에 있기때문에 그것은 버릴수 있다. p와 q의 4개의 모든 값이 검사된후 선을 버릴수 없으면 u_1과 u_2의 값으로부터 잘라 진 선의 끝점들을 결정한다.

```
#include "graphics. h"

#define ROUND (a) ( ( int) (a+0.5) )

int clipTest (float p, float q, float * ul, float * u2)
{
  float r;
  int retVal = TRUE;

  if (p < 0. 0) {
     r = q / p;
     if (r > *u2)
       retVal = FALSE;
     else
       if (r > *ul)
         *ul = r;
  }
  else
     if (p > 0. 0) {
       r = q / p;
```

```
        if (r < *ul)
          retVal = FALSE;
        else if (r < *u2)
          *u2 = r;
      }
      else
       /* p = 0, so line is parallel to this clipping edge */
        if (q < 0. 0)
         /* Line is outside clipping edge */
         retVal = FALSE;

    return (retVal);
  }

  void clipLine (dcPt winMin, dcPt winMax, wcPt2 pi, wcPt2 p2)
  {
    float ul = 0. 0, u2 = 1. 0, dx = p2. x - pl. x, dy;

    if (clipTest (-dx, pl. x - winMin. x, &ul, &u2))
      if (clipTest (dx, winMax. x - pl. x, &ul, &u2)) {
        dy = p2. y - pl. y;
        if (clipTest (-dy, pl. y - winMin. y, &ul, &u2))
          if (clipTest (dy, winMax. y - pl. y, &ul, &u2)) {
            if (u2 < 1. 0) {
              p2.x = pl.x + u2 * dx;
              p2.y = pl.y + u2 * dy;
            }
            if (ul > 0. 0) {
              pl.x += ul * dx;
              pi.y += ul * dy;
            }
            lineDDA (ROUND(pi.x), ROUND(pl.y), ROUND(p2.x),
                     ROUND(p2.y))
    }
  }
}
```

일반적으로 리앙-바스키알고리듬은 사귐점계산이 적기때문에 코헨-싸더랜드알고리듬보다 더 효과적이다. 파라메터 u_1과 u_2의 매 수정은 한번의 나누기만을 요구하며 선의 창문사귐점들은 u_1과 u_2의 마지막값이 계산되였을 때 한번만 계산된다. 반대로 코헨-싸더랜드알고리듬은 선경로를 따라 지어 선이 완전히 자르기창문밖에 있어도 사귐점을 반복적으로 계산할수 있다. 그리고 매개 사귐점계산은 나누기와 곱하기를 다같이 요구한다. 코헨-싸더랜드 및 리앙-바스키알고르듬들은 다 3차원자르기에 확장할수 있다(12장).

니콜-리-니콜선자르기

니콜-리-니콜(Nicholl-Lee-Nicholl 또는 NLN)알고리듬은 자르기창문주위에 더 많은 구역들을 만들어 개별적인 선분에 대한 여러번에 걸치는 자르기를 피한다. 실례로 코헨-싸더랜드방법에서는 사귐점이 자르기직4각형경계에 위치하거나 또는 완전히 버릴수 있는 선도 완전히 버리기전에 하나의 선경로를 따라서 다중사귐점계산이 진행될수 있다. 이 불필요한 사귐점계산을 NLN알고리듬에서는 사귐점위치를 계산하기전에 더 많은 구역검사를 진행함으로써 없앤다. 니콜-리-니콜알고리듬은 코헨-싸더랜드 및 리앙-바스키알고리듬에 비하여 비교와 나누기회수가 적다. 하지만 NLN알고리듬은 2차원자르기에만 적용될수 있다. 한편 리앙-바스키 및 코헨-싸더랜드방법들은 다같이 3차원장면에로 쉽게 확장된다.

끝점이 $\mathbf{P}_1$과 $\mathbf{P}_2$인 선에서 먼저 점 $\mathbf{P}_1$의 위치를 자르기직4각형에 대하여 있을수 있는 9개의 구역들에 대하여 결정한다. 실제로는 그림 6-10에 보여 준 3개의 구역들만이 고찰된다. $\mathbf{P}_1$이 다른 6개 구역중 어느 하나에 있으면 대칭변환을 리용하여 그림 6-10의 세개구역중 하나에로 옮길수 있다. 실례로 자르기창문 직접우의 구역은 선 $y=-x$에 대한 반사를 리용하거나 시계바늘반대방향 90°회전을 리용하여 자르기창문의 왼쪽 구역에로 변환할수 있다.

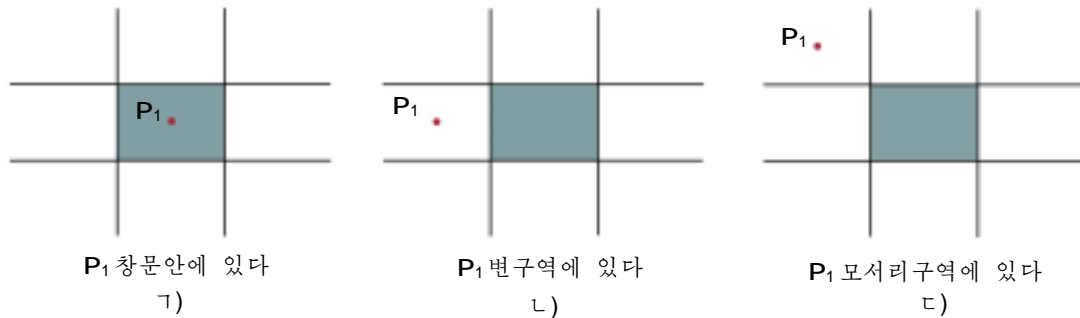



그림 6-10. NLN 선자르기알고리듬에서 선끝점 $\mathbf{P}_1$에 대한 3개의 가능한 위치

다음 $\mathbf{P}_1$에 대한 $\mathbf{P}_2$의 위치를 결정한다. 이렇게 하기 위하여 $\mathbf{P}_1$의 위치에 따라 평면에서 몇개의 새로운 구역들을 만든다. 새로운 구역들의 경계는 $\mathbf{P}_1$위치에서 시작하여 창문구석들을 통과하는 반직선이다. $\mathbf{P}_1$이 자르기창문안에 있고 $\mathbf{P}_2$가 밖에 있으면 그림 6-11에 보여 준 4개의 구역들이 설정된다. 다음에 4개의 구역(L,T,R,B)중 어느것이 $\mathbf{P}_2$를 포함하는가에 따라 해당한 창문경계와의 사귐점계산이 진행된다. 물론 $\mathbf{P}_1$과 $\mathbf{P}_2$가 다같이 자르기직4각형내부에 있으면 간단히 전체 선을 보관한다.

$\mathbf{P}_1$이 창문의 왼쪽 구역에 있으면 그림 6-12에 보여 준 4개의 구역 L, LT, LR, LB가 설정된다. 이 4개 구역들은 선분에 대하여 유일한 경계를 결정한다. 실례로 $\mathbf{P}_2$가 구역 L에 있으면 선을 왼쪽 경계에서 자르고 이 사귐점으로부터 $\mathbf{P}_2$까지의 선분을 보관한다. 그러나 $\mathbf{P}_2$가 구역 LT에 있으면 창문의 왼쪽 경계로부터 웃경계까지의 선분을 보관한다. $\mathbf{P}_2$가 4개 구역 L, LT, LR, LB중 어디에도 들어 가지 않으면 전체 선을 버린다.

세번째 경우 $\mathbf{P}_1$이 자르기창문의 왼쪽 우에 있을 때에는 그림 6-13의 자르기구역들을 리용한다. 이 경우 창문의 왼쪽 웃구석에 대한 $\mathbf{P}_1$의 위치에 따라 그림에서 보여 준 두개의 가능성이 있다. $\mathbf{P}_2$가 구역 T, L, TR, TB, LR, LB중 하나에 있으면 사귐점계산을 위한 유일한 자르기창문경계가 결정된다. 그렇지 않으면 전체 선은 버린다.

$\mathbf{P}_2$가 위치하고 있는 구역을 결정하기 위하여서는 선의 경사도를 자르기구역경계들의 경사도와 비교한다. 실례로 $\mathbf{P}_1$이 자르기직4각형의 왼쪽에 있는 경우(그림 6-12) $\mathbf{P}_2$는

$$\overline{\mathbf{P}_1\mathbf{P}_{TR}}\text{경사도} < \overline{\mathbf{P}_1\mathbf{P}_2}\text{경사도} < \overline{\mathbf{P}_1\mathbf{P}_{TL}}\text{경사도} \tag{6-14}$$

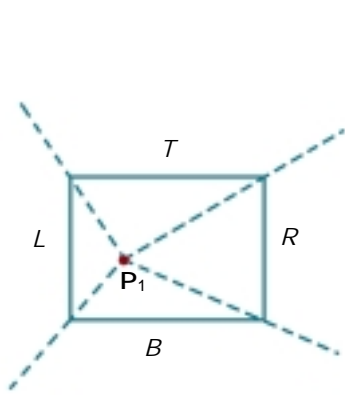



그림 6-11. $\mathbf{P}_1$이 자르기창문안에 있고 $\mathbf{P}_2$가 밖에 있을 때 NLN 알고리듬에서 리용하는 4개의 자르기구역

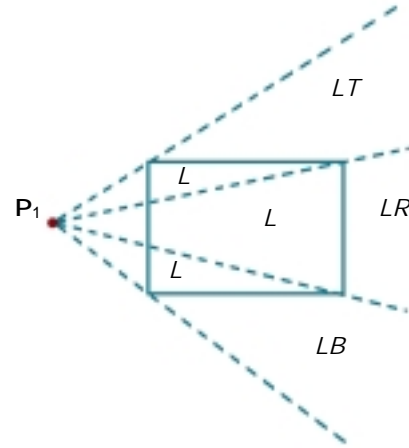



그림 6-12. $\mathbf{P}_1$이 자르기창문의 직접 왼쪽에 있을 때 NLN 알고리듬에서 리용하는 4개의 자르기구역

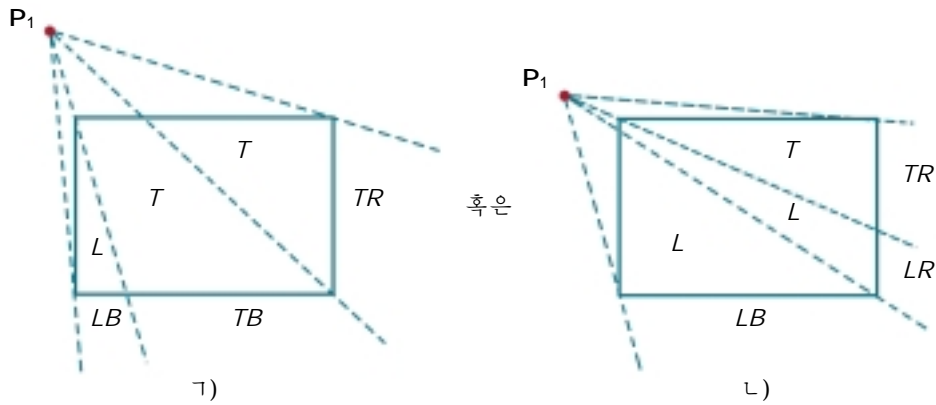



그림 6-13. P_1이 자르기창문의 왼쪽 우에 있을 때 NLN 알고리듬에서 리용하는 자르기구역들의 두가지 가능한 모임

또는

$$\frac{y_T - y_1}{x_R - x_1} < \frac{y_2 - y_1}{x_2 - x_1} < \frac{y_T - y_1}{x_L - x_1} \tag{6-15}$$

이면 구역 LT에 있다. 그리고

$$(y_T - y_1)(x_2 - x_1) < (x_L - x_1)(y_2 - y_1) \tag{6-16}$$

이면 전체 선을 버린다.

경사도검사에 리용된 자리표편차값과 적계산은 보관하여 사귐점계산에서도 리용한다. 보조변수방정식

$$x = x_1 + (x_2 - x_1)u$$
$$y = y_1 + (y_2 - y_1)u$$

로부터 창문의 왼쪽 경계에서 x사귐점위치는 $x=x_L$, $u=(x_L-x_1)/(x_2-x_1)$, 그러므로 y사귐점위치는

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_L - x_1) \tag{6-17}$$

이다. 그리고 웃경계에서 사귐점위치는 $y=y_T$, $u=(y_T-y_1)/(y_2-y_1)$이며

$$x = x_1 + \frac{x_2 - x_1}{y_2 - y_1}(y_T - y_1) \tag{6-18}$$

이다.

비직각자르기창문을 리용하는 선자르기

일부 응용들에서는 임의의 형태의 다각형에 대한 선자르기가 자주 제기된다. 리앙-바스키방법과 이전의 씨루스-벡크방법과 같이 선의 보조변수방정식에 기초한 알고리듬들은 볼록다각형창문에 쉽게 확장할수 있다. 이것은 알고리듬에 자르기구역의 경계에 대한 보조변수방정식들이 포함되도록 수정함으로써 이루어 진다. 선분들의 대체적인 선별은 자르기다각형의 자리표범위에서 선을 처리하면 될수 있다. 오목다각형자르기구역에 대하여서는 먼저 오목다각형을 볼록다각형들의 모임으로 분할해 놓으면 여전히 이 보조변수자르기절차를 적용할수 있다.

원 또는 다른 곡선경계자르기구역들도 역시 있을수 있지만 드물게 리용된다. 이런 구역들에 대한 자르기알고리듬은 비선형곡선방정식을 포함하기때문에 사귐점계산이 더디다. 첫 걸음에서 선들을 곡선자르기구역의 테두리직4각형(자리표범위)에 대하여 자를수 있다. 완전히 테두리직4각형의 밖이라고 식별되는 선들은 버린다. 내부선을 식별하기 위하여 원중심으로부터 선끝점들의 거리를 계산할수 있다. 선의 각 끝점까지의 거리의 두제곱이 다 반경의 두제곱보다 작거나 같으면 전체 선을 보관할수 있다. 남은 선들은 다음에 사귐점계산을 통하여 처리되는데 원과 선의 련립방정식을 풀어야 한다.

오목다각형의 분할

다각형둘레를 따라 차례로 련속된 변벡토르들의 벡토르적을 계산하면 오목다각형을 식별할수 있다. 일부 벡토르적들의 z성분은 정(+)인데 다른것들은 부(-)인 z성분을 가진다면 오목다각형이다. 그렇지 않으면 다각형은 볼록이다. 이것은 세개의 련속된 정점들이 같은 직선우에 있지 않다는것을 가정하고 있다. 같은 직선우에 있는 경우에는 이 정점들에 대한 두 변벡토르의 벡토르적은 령이다. 모든 정점들이 같은 직선우에 있으면 퇴화다각형(직선)으로 된다. 그림 6-14에서는 변벡토르들의 벡토르적을 리용하여 오목다각형을 식별하는 방법을 보여 주고 있다.

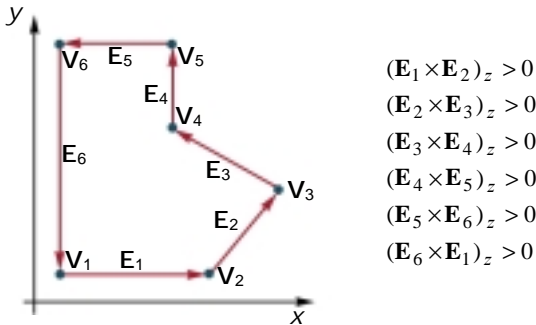



그림 6-14. 변벡토르들의 련속된 쌍의 벡토르적 계산에 의한 오목다각형 식별

xy평면에서 오목다각형을 분할하는 벡토르방법은 시계바늘반대방향순서로 변벡토르들의 벡토르적을 계산하고 벡토르적의 z성분의 부호를 주의하는것이다. 어떤 z성분이 부로 절환되면(그림 6-14에서와 같이) 다각형은 오목이며 다각형을 그 벡토르적쌍의 첫번째 변벡토르의 선을 따라 분할할수 있다. 다음의 실례에서 오목다각형분할방법을 설명한다.

실례 6-2. 오목다각형분할의 벡토르방법

그림 6-15는 6개의 변을 가지는 오목다각형을 보여 준다. 이 다각형에 대한 변벡토르들은

$$\mathbf{E}_1 = (1,0,0), \quad \mathbf{E}_2 = (1,1,0)$$
$$\mathbf{E}_3 = (1,-1,0), \quad \mathbf{E}_4 = (0,2,0)$$
$$\mathbf{E}_5 = (-3,0,0), \quad \mathbf{E}_6 = (0,-2,0)$$

와 같이 표현할수 있다. 여기서는 모든 변들이 xy평면상에 있기때문에 z성분은 0이다. 두개의 련속된 변벡토르에 대한 벡토르적 $\mathbf{E}_i \times \mathbf{E}_j$는 xy평면에 수직이고 z성분이 $\boldsymbol{E}_{ix}\boldsymbol{E}_{jy}-\boldsymbol{E}_{jx}\boldsymbol{E}_{iy}$와 같은 벡토르이다.

$$\mathbf{E}_1 \times \mathbf{E}_2 = (0,0,1), \quad \mathbf{E}_2 \times \mathbf{E}_3 = (0,0,-2)$$
$$\mathbf{E}_3 \times \mathbf{E}_4 = (0,0,2), \quad \mathbf{E}_4 \times \mathbf{E}_5 = (0,0,6)$$
$$\mathbf{E}_5 \times \mathbf{E}_6 = (0,0,6), \quad \mathbf{E}_6 \times \mathbf{E}_1 = (0,0,2)$$

벡토르적 $\mathbf{E}_2 \times \mathbf{E}_3$이 부의 z성분을 가지기때문에 다각형을 벡토르 $\mathbf{E}_2$의 선을 따라 분할한다. 이 변에 대한 선의 방정식은 경사도 1, y축자르기 -1을 가진다. 다음에 이 선과 다른 다각형변과의 사귐점을 결정하여 다각형을 두 쪼각으로 분할한다. 다른 변벡토르적들은 부가 아니므로 새로운 두 다각형들은 다 볼록이다.

회전방법으로도 오목다각형을 분할할수 있다. 다각형둘레를 시계바늘반대방향으로 돌면서 매개 다각형정점 V_k를 차례로 자리표원점에로 이동시킨다. 다음에 다음 정점 V_{k+1}이 x축에 놓이도록 시계바늘방향으로 회전시킨다. 이때 정점 V_{k+2}이 x축아래에 있으면 다각형은 오목이다. 그러면 다각형을 x축을 따라 두개의 새 다각형으로 분할하고 두개의 매 새 다각형에 대하여 오목검사를 반복한다. 그렇지 않으면 정점들을 x축에로 계속 회전시키면서 부의 y정점값에 대한 검사를 진행한다. 그림 6-16은 오목다각형을 분할하는 회전방법을 설명한다.

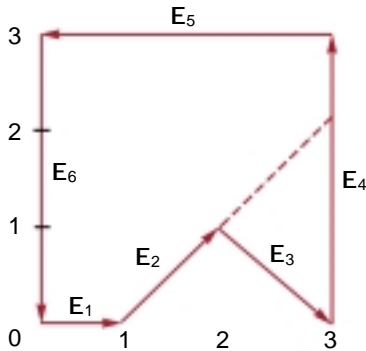



그림 6-15. 벡토르방법을 리용하여 오목다각형 분할

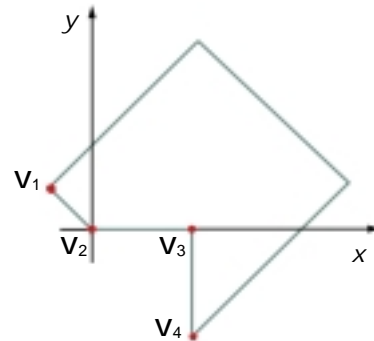



그림 6-16. 회전방법을 리용한 오목다각형분할 ($\mathbf{V}_3$을 x축에로 회전시키면 $\mathbf{V}_4$가 x축아래에 있다는것을 알게 된다. 그러므로 다각형을 선 $\overline{\mathbf{V}_2\mathbf{V}_3}$을 따라 분할한다.)

8절. 다각형자르기

다각형을 자르기 위하여서는 앞절에서 설명한 선자르기절차를 수정하면 된다. 선자르기에 의하여 처리된 다각형경계는 자르기창문에 대한 다각형의 방향에 따라 련결되지 않은 선분들로 현시될수 있다(그림 6-17). 실제로 현시하려는것은 자르기후에 그림 6-18에서와 같이 경계를 가진 구역이다. 다각형자르기를 하려면 하나 또는 그이상의 닫긴 구역들을 발생시키고 거기에 적당한 구역채우기를 진행하는 알고리듬이 있어야 한다. 다각형자르기의 출력은 잘라 진 다각형경계를 정의하는 정점들의 순서렬이여야 한다.

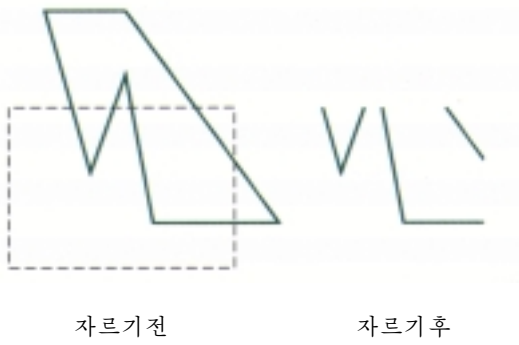



그림 6-17. 선자르기알고리듬으로 처리된 다각형의 현시

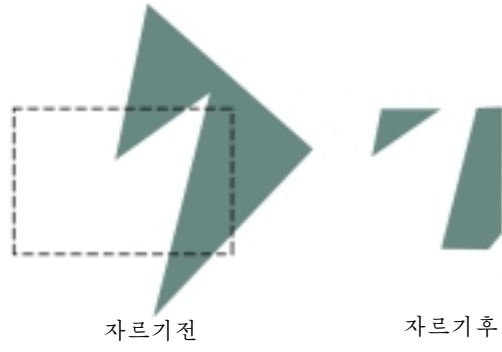



그림 6-18. 잘라 진 다각형의 정확한 현시

싸더랜드-호쥐맨다각형자르기

다각형의 모든 경계를 창문의 매개 경계에 대하여 처리하면 다각형을 정확히 자를수 있다. 이것은 다각형의 모든 정점들을 자르기직4각형의 매개 경계에 대하여 차례로 처리함으로써 얻을수 있다. 다각형정점들의 초기모임에서 시작하여 정점들의 새 순서렬을 만들기 위하여 먼저 다각형을 왼쪽 직4각형경계에 대하여 자른다. 정점들의 새 모임은 다음에 그림 6-19에서와 같이 계속하여 오른쪽 경계자르개, 아래경계자르개, 웃경계자르개에 통과시킨다. 매 걸음에서 출구정점들의 새 순서렬이 발생되며 다음번 창문경계자르개에 보낸다.

그림 6-19. 련속한 창문경계들에 대한 다각형자르기

다각형의 둘레를 따라 차례로 정점들을 처리할 때 4가지 가능한 경우가 있다. 린접한 다각형정점들의 매 쌍을 창문경계자르개에 통과시킬 때 다음의 검사처리를 진행한다. (1) 첫번째 정점이 창문경계밖에 있고 두번째 정점이 안에 있으면 다각형의 변과 창문경계와의 사귐점과 두번째 정점을 다같이 출력정점목록에 추가한다. (2) 두개의 입력정점이 다 창문경계안에 있으면 두번째 정점만 출력정점목록에 추가한다. (3) 첫번째 정점이 창문경계안에 있고 두번째 정점이 밖에 있으면 창문경계와의 경계사귐점만 출력정점목록에 추가한다. (4) 두개의 입력정점이 다 창문경계밖에 있으면 출력목록에 아무것도 추가하지 않는다. 다각형정점들의 련속된 쌍에 대한 이 4가지 경우들이 그림 6-20에서 설명된다. 일단 모든 정점들이 자르기창문의 하나의 경계에 대하여 다 처리되였으면 정점들의 출력목록은 창문의 다음번 경계에 대하여 자르기하게 된다.

이 방법을 창문의 왼쪽 경계에 대한 그림 6-21의 구역의 처리를 가지고 설명하자. 정점 1과 2는 경계밖에 있다. 정점 3에로 이동하면 그것은 안에 있다. 사귐점을 계산하여 사귐점과 정점 3을 다 같이 보관한다. 정점 4와 5는 내부에 있다. 그것들 역시 보관한다. 마지막 여섯번째 정점은 외부에 있으며 따라서 사귐점을 찾아 보관한다. 보관된 5개의 점들을 가지고 창문의 다음번 경계에 대해 처리를 반복한다.

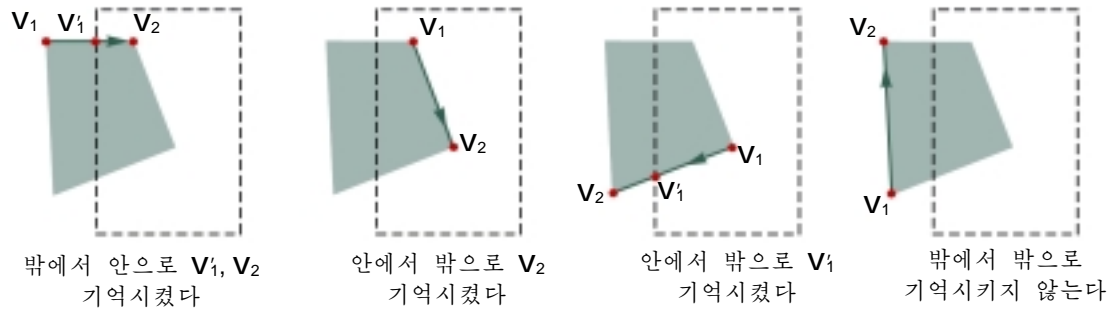



그림 6-20. 창문의 왼쪽 경계에 대한 다각형정점쌍의 련속된 처리

우에서 설명한 알고리듬을 실현하자면 다각형이 창문의 매 경계에 의하여 잘라 질 때 출력정점목록에 대한 기억기설정이 요구된다. 매 걸음에서 개별적인 점들을 간단히 자르고 잘라 진 정점들을 다음번 경계자르개에 통과시키면 중간출력정점목록을 없앨수 있다. 이것은 병렬처리기로 또는 하나의 처리기와 자르기루틴의 관흐름으로 해결할수 있다. 점(입력정점 또는 계산된 사귐점)은 그것이 4개의 모든 경계자르개에 의해 내부에 있다거나 또는 창문경계에 있다는것이 결정된후에야만 출력정점목록에 추가된다. 그렇지 않으면 점은 관흐름을 계속하지 않는다. 그림 6-22는 다각형과 자르기창문의 사귐점들을 보여 준다. 그림 6-23에서는 경계자르개의 관흐름을 통한 그림 6-22의 다각형정점들의 전진을 보여 준다.

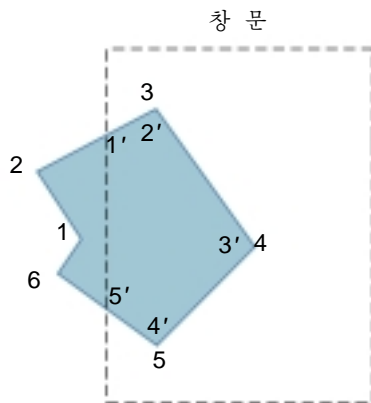



그림 6-21. 정점 1에서 시작하는 창문의 왼쪽 경계에 대한 다각형자르기(어깨표식이 붙은 번호는 이 창문경계에 대한 출력정점목록에 들어 가는 점들이다.)

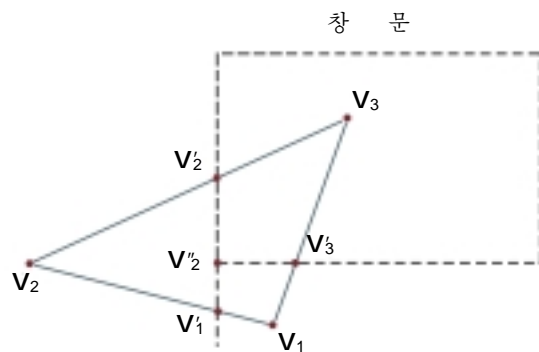



그림 6-22. 직4각형자르기창문과 겹치는 다각형

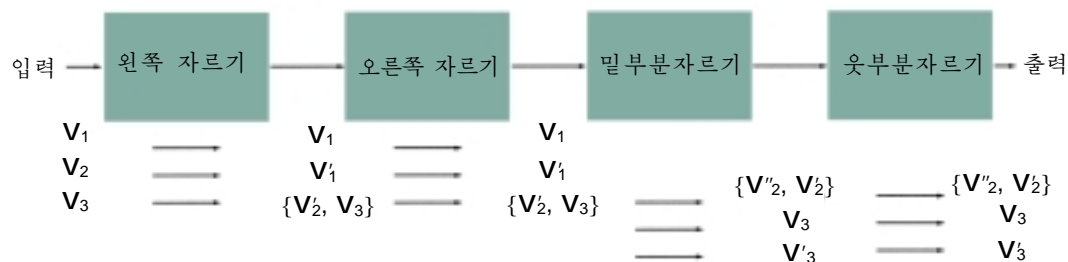



그림 6-23. 경계자르기 관흐름을 통한 그림 6-22의 다각형정점들의 처리(모든 정점들이 관흐름을 통해 처리된후의 잘라 진 다각형에 대한 정점목록은 $\{V''_2, V'_2, V_3, V'_3\}$이다.)

다음의 프로그람은 관흐름자르기절차를 보여 준다. 배렬 s에는 자르기창문의 매 경계에 대하여 잘라 진 제일 마지막점을 기록한다. 주루틴은 매개 정점 p를 창문의 첫번째 경계에 대한 자르기를 위하여 clipPoint루틴에 넘겨 준다. 끝점 p와 s[boundary]에 의해 정의되는 선이 창문의 이 경계를 가로지르면 사귐점이 계산되고 다음번 자르기단계에 넘겨 진다. p가 창문내부에 있으면 그것은 다음번 자르기단계에 넘어 간다. 창문의 모든 경계에 대한 자르기에서 살아 남은 점들만 점들의 출력배렬에 들어간다. 배렬 firstPoint에는 창문의 매 경계의 그 경계에 대하여 잘라 진 첫번째 점을 기억시킨다. 다각형의 모든 정점들이 다 처리된후 닫기루틴은 매 경계에 대하여 잘라 진 첫 점과 마지막점으로 정의되는 선을 자른다.

```
typedef enum { Left, Right, Bottom, Top } Edge;
#define N_EDGE 4

int inside (wcPt2 p, Edge b, dcPt wMin, dcPt wMax)
{
  switch (b) {
  case Left: if (p.x < wMin.x) return (FALSE); break;
  case Right: if (p.x > wMax.x) return (FALSE); break;
  case Bottom: if (p.y < wMin.y) return (FALSE); break;
  case Top: if (p.y > wMax.y) return (FALSE); break;
  }
  return (TRUE) ;
}

int cross (wcPt2 pi, wcPt2 p2, Edge b, dcPt wMin, dcPt wMax)
{
  if (inside (pi, b, wMin, wMax) == inside (p2, b, wMin, wMax))
     return (FALSE) ;
  else return (TRUE) ;
}

wcPt2 intersect (wcPt2 pi, wcPt2 p2, Edge b, dcPt wMin, dcPt wMax)
{
  wcPt2 iPt;
  float m;

  if (pl.x != p2.x) m = (pl.y - p2.y) / (pl.x - p2.x);
  switch (b) {
  case Left:
    iPt.x = wMin.x;
    iPt.y = p2.y + (wMin.x - p2.x) * m;
    break;
  case Right:
    iPt.x = wMax.x;
    iPt.y = p2.y + (wMax.x - p2.x) * m;
    break;
```

```
    case Bottom:
      iPt.y = wMin.y;
      if (pl.x != p2.x) iPt.x = p2.x + (wMin.y - p2.y) / m;
      else iPt.x = p2.x;
      break;
    case Top:
      iPt.y = wMax.y;
      if (pl.x != p2.x) iPt.x = p2.x + (wMax.y - p2.y) / m;
      else iPt.x = p2.x;
      break;
   }
   return (iPt) ;
}

void clipPoint (wcPt2 p, Edge b, dcPt wMin, dcPt wMax,
          wcPt2 * pOut, int * cnt, wcPt2 * first[], wcPt2 * s)
{
 wcPt2 iPt;

 /* If no previous point exists for this edge, save this point. */
 if (!first[b])
   first [b] = &p;
 else
   /* Previous point exists. If 'p' and previous point cross edge,
      find intersection. Clip against next boundary, if any. If
      no more edges, add intersection to output list. */
   if (cross (p, s[b], b, wMin, wMax)) {
      iPt = intersect (p, s[b], b, wMin, wMax) ;
      if (b < Top)
        clipPoint (iPt, b+1, wMin, wMax, pOut, cnt, first, s);
      else {
        p0ut[*cnt] = iPt; (*cnt)++;
      }
   }

 s[b] = p;    /* Save 'p' as most recent point for this edge */

 /* For all, if point is 'inside' proceed to next clip edge, if any */
 if (inside (p, b, wMin, wMax))
   if (b < Top)
     clipPoint (p, b+1, wMin, wMax, pOut, cnt, first, s);
   else {
     p0ut[*cnt] = p; (*cnt)++;
   }
}

void closeClip (dcPt wMin, dcPt wMax, wcPt2 * pOut,
```

```
                    int * cnt, wcPt2 * first[], wcPt2 * s)
{
  wcPt2 i ;
  Edge b;

  for (b = Left; b <= Top; b++) {
    if (cross (s[b], *first[b], b, wMin, wMax)) {
      i = intersect (s[b], *first[b], b, wMin, wMax);
      if (b < Top)
        clipPoint (i, b+1, wMin, wMax, pOut, cnt, first, s);
      else {
        pOut[*cnt] = i; (*cnt)++;
      }
    }
  }
}

int clipPolygon (dcPt wMin, dcPt wMax, int n, wcPt2 * pIn, wcPt2 * pOut)
{
  /* 'first' holds pointer to first point processed against a clip
   edge. 's' holds most recent point processed against an edge */
 wcPt2 * first[N_EDGE] = { 0, 0, 0, 0 }, s [N_EDGE] ;
 int i, cnt = 0;

 for (i=0; i<n; i++)
   clipPoint (pIn[i]. Left, wMin, wMax, pOut, &cnt, first, s);
 closeClip (wMin, wMax, pOut, &cnt, first, s) ;
 return (cnt);
}
```

볼록다각형은 싸더랜드-호쥐맨알고리듬에 의하여 정확히 잘라 지지만 오목다각형은 그림 6-24에서 보여 주는바와 같이 인연이 없는 선들이 현시될수 있다. 이것은 잘라 진 다각형이 둘 또는 그이상의 분리부분들을 가질 때 일어 난다. 그런데 하나의 출력정점목록만 있으므로 항상 목록에서 마지막정점이 첫번째 정점에 련결된다. 오목다각형을 정확히 현시할수 있는 여러가지 방법들이 있다. 하나는 오목다각형을 둘 또는 그이상의 볼록다각형들로 분할하고 매개 볼록다각형을 따로따로 처리하는것이다. 다른 가능성은 어떤 자르기창문경계를 따르는 다중정점들에 대하여 마지막정점목록을 검사하여 정점들의 쌍이 정확히 련결되도록 싸러랜드-호쥐맨방법을 수정하는것이다. 마지막으로 다음 부분에서 서술되는 웨일러-아써톤알고리듬 또는 웨일러알고리듬과 같이 보다 일반적인 다각형자르개를 리용할수 있다.

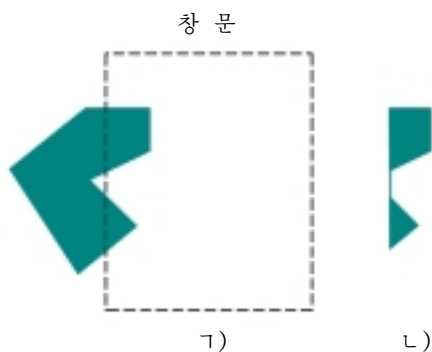



그림 6-24. 싸더랜드-호쥐맨자르개에 의한 (ㄱ)의 오목다각형자르기는 (ㄴ)에서 두개의 련결된 구역을 만든다.

웨일러-아써톤다각형자르기

여기서는 창문경계들에 대한 정점처리절차를 오목다각형이 정확히 현시되도록 수정한다. 이 자르기절차는 보이는면식별방법으로 개발되였으므로 임의의 다각형자르기구역에 적용할수 있다.

이 알고리듬의 기초적인 사상은 정점들을 처리할 때 항상 다각형변을 따라 처리하는것이 아니라 때때로 창문경계를 따르게 하는 것이다. 어느 경로를 따르는가 하는것은 다각형의 처리방향(시계바늘 또는 시계바늘반대)과 현재 처리되고 있는 다각형정점쌍이 외부-내부쌍인가 또는 내부-외부쌍을 표현하는가에 하는데 관계된다. 다각형정점들의 시계바늘방향처리에 대하여서는 다음의 규칙을 리용한다.

- 정점들의 외부-내부쌍에 대하여서는 다각형변을 따른다.
- 정점들의 내부-외부쌍에 대하여서는 시계바늘방향으로 창문경계를 따른다.

그림 6-25에서 직4각형자르기창문에 대한 웨일러-아써톤알고리듬에서의 처리방향과 얻어 진 잘라 진 다각형을 보여 준다.

웨일러-아써톤알고리듬을 개선한것이 웨일러알고리듬이며 그것은 임의의 다각형을 임의의 다각형자르기구역에 대해 자르는데 립체구성기하개념을 적용하고 있다. 그림 6-26은 이 방법의 일반적인 사상을 설명한다. 이 그림의 두개의 다각형에 대하여 정확히 잘라 진 다각형은 자르기다각형과 다각형물체의 사귐으로 계산된다.

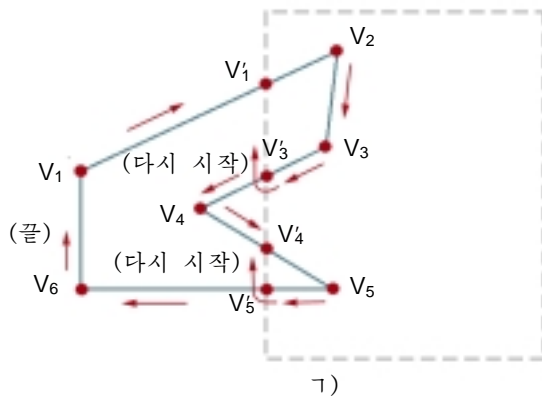



그림 6-25. 웨일러-아써톤알고리듬에 의해 오목다각형 ㄱ를 자르면 ㄴ의 두개의 분리된 다각형구역이 발생된다.

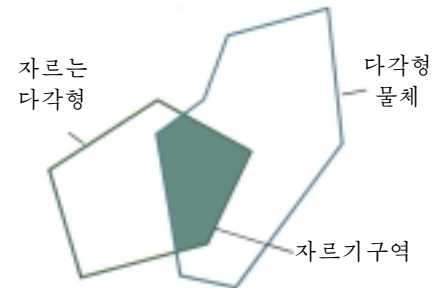



그림 6-26. 두 다각형구역의 사귐을 결정하는것에 의한 다각형자르기

기타 다각형자르기알고리듬

여러가지 보조변수선자르기방법들은 다각형자르기에도 적용되고 있다. 특히 볼록다각형자르기창문에 대한 자르기에 많이 적용되고 있다. 실례로 리앙-바스키선자르기방법은 싸더랜드-호쥐맨방법에서와 류사한 방식으로 다각형자르기에 확장될수 있다. 선의 보조변수표현은 선자르기에서 리용한것과 류사한 구역검사절차를 리용하여 다각형둘레를 따라 차례로 다각형변들을 처리하는데 리용된다.

9절. 곡선자르기

곡선경계를 가지는 구역들은 앞에서 설명한것과 류사한 방법들에 의해 자를수 있다. 그러나 곡선자르기절차는 비선형방정식을 처리해야 하며 이것은 선형경계를 가지는 물체보다 더 많은 처리를 요구한다.

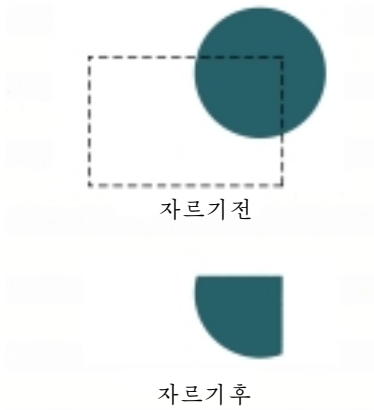



그림 6-27. 채워 진 원의 자르기

먼저 원 또는 다른 곡선물체의 테두리직4각형을 직4각형자르기창문과의 겹침검사에 리용할수 있다. 물체의 테두리직4각형이 완전히 창문안에 있으면 물체를 보관한다. 직4각형이 완전히 창문밖에 있다고 결정되면 그 물체는 버린다. 이 두 경우에는 앞으로 더 필요한 계산이 없다. 그러나 테두리직4각형검사가 실패하면 다른 계산보관방법을 써야 한다. 원에 대하여서는 곡선창문사귐을 계산하기전에 예비검사를 위하여 개별적인 4분구 또는 8분구의 자리표범위를 리용할수 있다. 타원에 대하여서는 개별적인 4분구의 자리표범위를 검사할수 있다. 그림 6-27은 직4각형창문에 대한 원자르기를 설명한다.

이러한 절차들은 일반적으로 다각형자르기구역에 대한 곡선물체들의 자르기에 적용될수 있다. 첫번째 통과에서 물체의 테두리직4각형을 자르기구역의 테두리직4각형에 대하여 자르게 된다. 두 구역이 겹치면 자르기사귐점을 얻기 위하여 련립선-곡선방정식을 풀어야 한다.

10절. 본문자르기

도형처리프로그람들에서 리용할수 있는 여러가지 본문자르기수법들이 있다. 리용되는 자르기수법은 문자발생에 리용된 방법과 개별적인 응용의 요구에 관계된다.

창문경계에 대하여 문자렬을 처리하는 제일 간단한 방법은 그림 6-28에 보여 준 전체를 주지 않으면 그만둔다는 문자렬자르기전략을 리용하는것이다. 문자렬전체가 자르기창문안에 있으면 그것은 보존한다. 그렇지 않으면 그 문자렬은 버린다. 이 절차는 본문무늬주위의 테두리직4각형을 조사하여 실현한다. 이때 직4각형의 경계위치가 창문경계와 비교되며 겹침이 조금이라도 있으면 그 문자렬은 다 버린다. 이 방법은 제일 빠른 본문자르기이다.

그림 6-28. 전체 문자렬에 대한 테두리직4각형을 리용하는 본문자르기

창문경계와 겹치는 문자렬을 버리는 또 한가지는 전체를 주지 않으면 그만둔다는 문자자르기전략을 리용하는것이다. 여기서는 완전히 창문안에 있지 않는 문자들만을 버린다(그림 6-29). 이 경우 개별적인 문자의 경계범위가 창문과 비교된다. 창문경계와 겹치거나 또는 밖에 있는 모든 문자는 잘라 진다.

본문자르기처리의 마지막방법은 개별적인 문자의 구성요소를 자르는것이다. 이때에는 선을 취급할 때와 대단히 류사한 방법으로 문자를 취급한다. 개별적인 문자가 자르기창문경계와 겹치면 창문밖에 있는 문자부분을 잘라 버린다(그림 6-30). 선분들로 형성되는 륜곽선문자폰트는 선자르기알고리듬을 리용하여 이 방식으로 처리할수 있다. 비트넘기기로 정의되는 문자들은 문자격자무늬에서 개별적인 화소들의 상대위치를 자르기경계와 비교하여 자를수 있다.

11절. 외부보존자르기

지금까지는 자르기구역밖의 모든것을 없애버리는 구역내부보존그림자르기절차만을 고찰하였다. 이 절차들에 의하여 보관되는것은 구역내부이다. 일부 경우들에서는 반대로 할수도 있다. 즉 지적된 구역의 외부를 보존하게 그림을 자를 필요가 있다. 보관될 그림부분들은 구역밖에 있는것들이다. 이것을 **외부보존자르기**(exterior clipping)라고 한다.

그림 6-29. 개별적인 문자의 테두리 직4각형을 리용하는 본문자르기

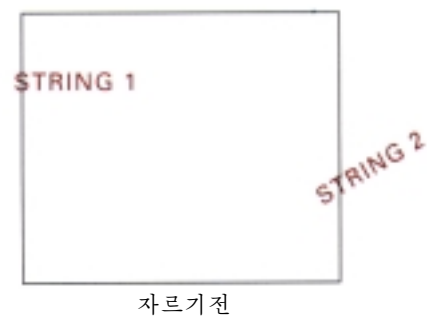
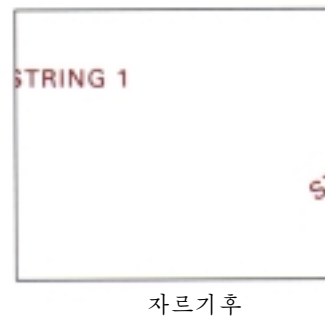
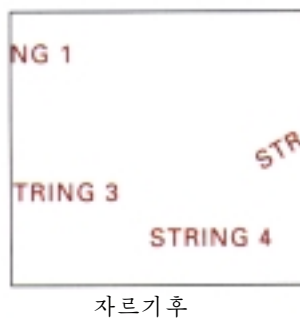



그림 6-30. 개별적인 문자의 구성요소에 대하여 수행되는 본문자르기

외부보존자르기응용의 대표적인 실례는 다중창문체계이다. 화면창문을 정확히 현시하기 위하여 자주 내부 및 외부보존자르기를 다같이 적용하여야 한다. 그림 6-31은 다중창문화면을 보여 준다. 창문의 물체들은 그 창문의 내부보존으로 자른다. 다른 더 높은 우선권창문이 이 물체와 겹칠 때에는 물체가 다시 겹치는 창문의 외부보존으로 잘라 진다.

외부보존자르기는 또한 겹침그림을 요구하는 다른 응용들에서도 리용된다. 이런 실례는 광고 또는 출판응용에서 페지편성설계 또는 그림에 표식이나 설계무늬를 추가하는 경우들이다. 이 기술은 또한 그라프, 지도 또는 도식을 결합하는데도 리용될수 있다. 이 응용들에서는 더 큰 그림에 삽입을 위한 공간을 제공하기 위하여 외부보존자르기를 리용할수 있다.

오목다각형창문의 내부를 보존하는 물체자르기절차에도 외부보존자르기를 리용할수 있다. 그림 6-32는 정점 $V_1V_2V_3V_4V_5$를 가지는 오목다각형창문의 내부보존으로 잘라 질 직선 $\overline{P_1P_2}$ 을 보여 준다. 선 $\overline{P_1P_2}$ 를 두번의 통과로 자른다. (1) 먼저 $\overline{P_1P_2}$ 을 볼록다각형 $V_1V_2V_3V_4$의 내부보존으로 잘라 토막 $\overline{P_1'P_2'}$ 를 얻는다(그림 6-32 ㄴ). (2) 그다음 얻어 진 토막 $\overline{P_1'P_2'}$ 을 볼록다각형 $V_1V_5V_4$의 외부보존으로 잘라 선분 $\overline{P_1''P_2'}$ 를 얻는다.

그림 6-31. 내부 및 외부보존 자르기실레를 다같이 보여 주는 다중창문화면현시

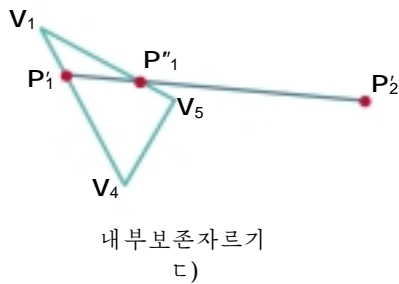
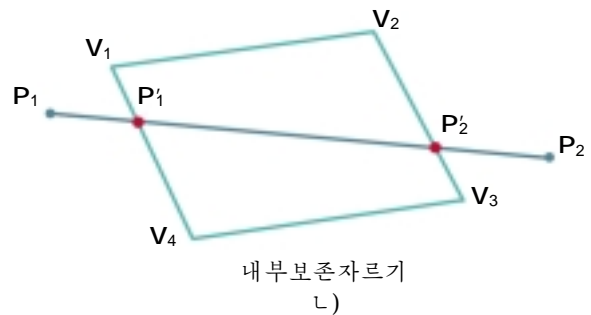
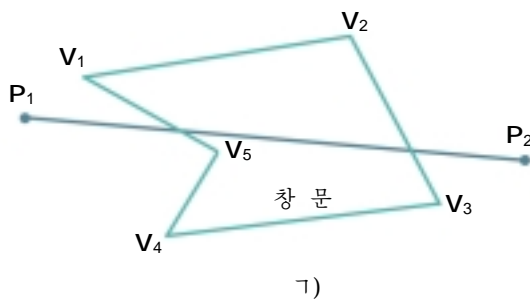



그림 6-32. 정점이 $V_1V_2V_3V_4V_5$인 오목다각형(ㄱ)에 대한 선분 $\overline{P_1P_2}$ 의 내부보존자르기(잘라 진 선 $\overline{P_1''P_2'}$ 을 얻기 위하여 볼록다각형 $V_1V_2V_3V_4$(ㄴ)와 볼록다각형 $V_1V_5V_4$(ㄷ)를 리용한다.)

요약

이 장에서는 2차원세계자리표의 장면을 어떻게 현시장치에 넘길수 있는가를 보았다. 보기변환과정의 흐름에는 모형화변환을 리용한 세계자리표의 장면만들기, 세계자리표의 보기자리표에로의 이동, 보기자리표의 물체서술을 정규화된 장치자리표로 넘기기, 마지막으로 장치자리표에로의 넘기기가 들어 간다. 정규화된 자리표는 0부터 1사이의 구간에서 정의되며 이것은 보기프로그람들을 개별적인 출력장치와 독립시키는데 쓰인다.

보기자리표는 보기원점의 세계자리표위치, 보기의 y축방향을 정의하는 보기벡토르를 가지고 지적한다. 이 파라메터들은 세계자리표의 물체서술을 보기자리표로 넘기는 보기변환행렬을 만드는데 쓰인다.

그다음 창문을 보기자리표로 설정하고 보임창을 정규화된 장치자리표로 지적한다. 일반적으로 창문과 보임창은 표준위치의 직4각형이다(직4각형의 경계들이 자리표축들에 평행이다). 보기자리표로부터 정규화된 장치자리표에로의 넘기기는 창문에서의 상대적위치들이 보임창에서 유지되도록 수행된다.

도형처리프로그람작성패키지들에서 보기함수들은 하나 혹은 그이상의 보기파라메터들의 모임을 만드는데 사용된다. 한 함수는 일반적으로 세계자리표를 보기자리표로 변환하기 위한 행렬의 원소들을 계산하기 위한것이다. 다른 함수는 창문-보임창변환행렬을 설정하는데 사용되며 3번째 함수는 보기변환들의 결합을 지적하고 창문을 보기표에 넘기는데 사용되게 된다. 보기표에 적혀 있는 개별적인 함수들을 지적하면 서로 다른 각이한 보기결합을 선택할수 있다.

자르기파라메터들이 off로 설정되여 있지 않는한 물체들이 출력장치에 현시될 때 창문(또는 보임창)밖의 장면의 모든 부분들은 잘라 버린다. 많은 프로그람들에서 자르기는 정규화된 장치자리표에서 수행된다. 그러므로 모든 변환들이 자르기알고리듬을 적용하기전에 하나의 변환연산들로 결합될수 있게 한다. 자르기구역을 일반적으로 자르기창문이라고 한다. 혹은 창문과 보임창이 표준직4각형인 경우는 자르기직4각형이라고 한다. 자르기창문경계에 대하여 물체를 자르기 위한 여러가지 알고리듬들이 개발되였다.

선자르기알고리듬에는 코헨-싸더랜드방법, 리앙-바스키방법, 니콜-리-니콜방법이 있다. 코헨-싸더랜드방법이 널리 리용되고 있다. 왜냐하면 그것은 첫 선자르기알고리듬의 하나로 개발된것이기때문이다. 구역코드들이 자르기창문경계인 직4각형에 대한 선의 끝점들의 상대적위치를 식별하는데 쓰인다. 완전히 창문안에 놓인다거나 완전히 창문밖에 놓인다는것을 단번에 식별할수 없는 선들은 창문경계들에 대하여 잘라 지게 된다. 리앙과 바스키는 씨루스-벡크의 알고리듬에서와 류사하게 직선의 보존변수표현을 써서 사귐점계산을 줄이는 더 효과적인 선자르기절차를 세웠다. 니콜-리-니콜 알고리듬은 사귐점계산을 더욱 줄이기 위하여 평면에서 더 많은 검사구역들을 사용하고 있다. 보조변수직선자르기는 오목자르기다각형과 3차원자르기창문에 쉽게 확장된다.

선자르기는 오목다각형자르기창문과 곡선경계를 가지는 자르기창문에 대하여서도 수행될수 있다. 오목다각형에서는 오목다각형을 몇개의 볼록다각형으로 분할하는 벡토르방법이나 회전방법을 사용할수 있다. 곡선자르기창문에서는 곡선의 방정식을 써서 선의 사귐점을 계산한다.

다각형자르기알고리듬으로서는 싸더랜드-호쥐맨방법, 리앙-바스키방법, 웨일리-아세톤방법이 있다. 싸더랜드-호쥐맨방법에서는 볼록다각형의 정점들을 직4각형창문의 4개의 경계에 대하여 차례로 처리하여 잘라 진 다각형에 대한 출력정점목록을 만든다. 리앙과 바스키는 보조변수직선방정식을 써서 볼록다각형의 변들을 표현하고 선자르기에서와 류사한 시험을 진행하여 잘라 진 다각형에 대한 출력정점목록을 만든다. 웨일리-아써톤방법과 웨일리방법은 볼록, 오목다각형을 다같이 정확하게 자른다. 이

자르기방법들은 일반다각형자르기창문에도 쓸수 있다. 웨일리-아써톤알고리듬은 다각형정점들을 차례로 처리하여 하나 혹은 그이상의 출력다각형정점목록을 만든다. 웨일리방법은 두개 다각형들의 사귐구역을 조사하여 자르기를 수행한다.

곡선경계를 가지는 물체들은 직4각형자르기창문에 대하여 곡선의 방정식을 리용하여 사귐점을 처리한다. 이 자르기 절차들은 직선자르기나 다각형자르기 보다 뜨다. 왜냐하면 곡선의 방정식은 비선형이기때문이다.

제일 빠른 본문자르기방법은 문자렬의 일부분이라도 임의의 창문경계밖에 놓이면 문자렬을 전부 잘라 버리는것이다. 본문자르기의 다른 방법은 모두가 아니면 그만두는 방법을 문자렬의 개별적인 문자들에 적용하는것이다. 세번째 방법은 문자들이 점격자로 정의되는가 륜곽선폰트로 정의되는가에 따라 점, 선, 다각형, 원자르기를 문자렬의 개별적문자들에 적용하는것이다.

그림을 삽입하거나 다중화면창문들을 관리하는것과 같은 일부 응용들에서는 외부보존자르기가 진행된다. 이 경우 창문안에 있는 장면의 모든 부분들이 잘라 지고 바깥부분들은 보관된다.

참고문헌

선자르기알고리듬들은 Sproull과 Sutherland(1968), Cyrus와 Beck(1978), Liang과 Barsky(1984)에서 고찰되고 있다. 코헨-싸더랜드선자르기알고리듬의 속도를 높이기 위한 방법들을 Duvanenko(1990)에서 주고 있다.

다각형자르기방법들은 Sutherland와 Hodgeman(1974)과 Liang과 Barsky(1983)에서 설명하였다. 임의의 모양의 다각형의 임의의 모양의 자르기창문에 대한 자르기의 일반적수법들은 Weiler와 Atherton(1977)과 Weiler (1980)에서 주고 있다.

PHIGS에서의 2차원보기연산들은 Howard들(1991), Gaskins(1992), Hopgood와 Duce(1991), Blake(1993)에서 설명하였다. GKS보기연산들에 대한 정보는 Hopgood(1983)들과 Enderle(1984)들을 보시오.

련습문제

6-1. 보기자리표원점 $\mathbf{P}_0$과 보기벡토르 **V**를 주고 세계자리표를 보기자리표로 변환하는 행렬의 원소들을 계산하기 위한 evaluateViewOrientationMatrix함수를 실현하는 프로그람을 쓰시오.

6-2. 창문-보임창변환식 6-3을 처음에 창문을 보임창크기로 비례변환하고 그다음 비례변환된 창문을 보임창위치에 옮기는 방법으로 유도하시오.

6-3. 창문-보임창변환을 수행하는 행렬의 원소들을 계산하기 위한 evaluateViewMapping Matrix함수를 실현하는 프로그람을 쓰시오.

6-4. 보기표에서 보기첨수로 참조되며 viewMatrix와 viewMappingMatrix를 결합하여 결과를 기억시키는 setViewRepresentation함수를 실현하는 프로그람을 쓰시오.

6-5. 자르기와 워크스테이션변환이 없이 보기의 관흐름을 실현하는 프로그람을 쓰시오.이 프로그람은 장면을 모형화자리표변환, 보기자리표계, 창문-보임창쌍의 지적으로 만들수 있게 되여야 한다. 여러가지 보기변환파라메터 모임을 기억시키기 위하여 선택항목으로서 보기표를 실현할수 있다.

6-6. 워크스테이션변환에 대한 행렬표현을 유도하시오.

6-7. 자르기는 없지만 워크스테이션변환은 있는 보기관흐름을 실현하는 프로그람들의 모임을 쓰시오. 이 프로그람들은 장면을 모형화자리표변환, 지적된 보기자리표계, 지적된 창문-보임창쌍, 워크스테이션변환파라메터로 만들수 있게 하여야 한다. 지적된 세계자리표장면에 대하여 합성보기변환행렬은 장면을 현시장치의 출력으로 변환되게 하여야 한다.

6-8. 코헨-싸더랜드원자르기알고리듬을 실현하시오.

6-9. 리앙-바스키선자르기알고리듬에서 사귐점에 파라메터 u_1과 u_2를 계산하기 위한 여러가지 시험과 방법들의 배후의 추리를 주의깊이 고찰하시오.

6-10. 자르기창문에 대하여 여러가지 각이한 방향을 가지는 선에 대한 코헨-싸더랜드와 리앙-바스키선자르기알고리듬에서 집행되는 산수연산의 회수를 비교하시오.

6-11. 리앙-바스키선자르기알고리듬을 실현하는 프로그람을 쓰시오.

6-12. 그림 6-10의 3개 구역에 대한 사귐점계산을 xy평면의 기타 6개 구역에로 넘기기 위한 대칭변환을 창안하시오.

6-13. 선끝점들의 임의의 입력쌍에 대한 니콜-리-니콜선자르기의 구체적인 알고리듬을 짜시오.

6-14. NLN알고리듬에서 수행되는 산수연산의 회수를 자르기창문에 대하여 여러가지 각이한 방향의 선에 대하여 코헨-싸더랜드알고리듬, 리앙-바스키선자르기알고리듬들과 비교하시오.

6-15. 변벡토르의 쌍들의 벡토르적을 계산하여 오목다각형을 식별하는 루틴을 쓰시오.

6-16. 벡토르방법을 써서 오목다각형을 분할하는 루틴을 쓰시오.

6-17. 회전방법을 써서 오목다각형을 분할하는 루틴을 쓰시오.

6-18. 리앙-바스키선자르기알고리듬을 다각형자르기에 적응시키시오.

6-19. 자르기창문이 표준위치의 직4각형이라고 가정하고 웨일리-아세톤다각형자르기에 대한 구체적인 알고리듬을 짜시오.

6-20. 자르기창문이 임의로 지적되는 다각형일 때 웨일리-아세톤다각형자르기에 대한 알고리듬을 창안하시오.

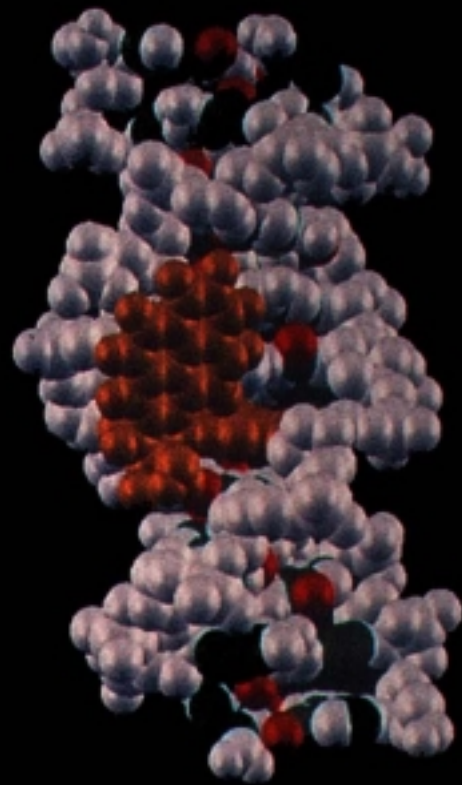
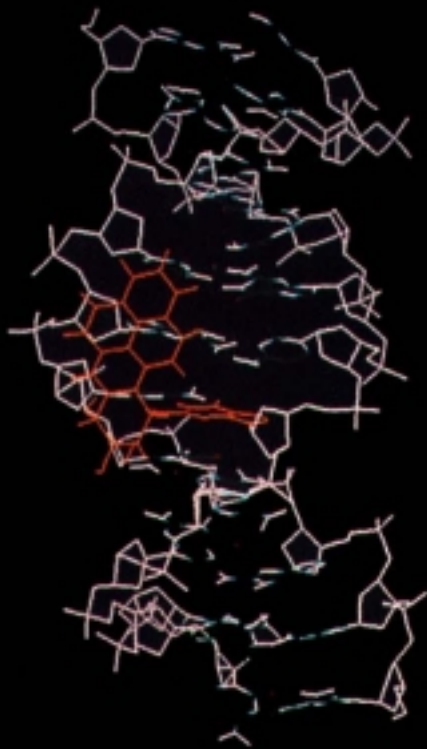
6-21. 직4각형창문에 대한 타원자르기루틴을 쓰시오.

6-22. 본문에서 모든 문자들이 같은 너비를 가진다고 가정하고 모두가 아니면 그만두는 문자자르기전략에 기초하여 문자렬을 자르는 본문자르기알고리듬을 전개하시오.

6-23. 문자들이 지적된 크기의 화소격자로 정의되였을 때 개별적인 문자들을 자르는 본문자르기알고리듬을 전개하시오.

6-24. 정의된 그림의 임의의 부분에 대하여 임의로 지적되는 창문을 써서 외부보존자르기를 실현하는 루틴을 쓰시오.

6-25. 분할창문체계의 화면에서 내부보존과 외부보존자르기를 수행하는 루틴을 쓰시오. 루틴에 대한 입력은 화면에서 창문위치들의 모임, 매 창문에서 현시될 물체, 창문의 우선권들이다. 매개의 물체들은 각기 자기 창문에 맞게 잘라 져야 한다. 더 높은 현시준위의 창문과 겹치는 부분들은 잘라 버려야 한다.

7장. 구조물 및 계층적인 모형화

대단히 많은 응용분야들에서 그림의 다른 부분들에 영향을 미침이 없이 그림의 개별적인 부분들을 만들고 처리할수 있으면 편리하다. 이 능력을 대부분의 도형처리프로그람들에서 여러가지 형식으로 제공해 주고 있다. 그림안의 매개 물체를 개별적모듈로 정의하는 능력이 있으면 그림에 대한 수정을 더 쉽게 할수 있다. 설계분야들에서는 그림의 다른 부분들을 다치지 않고 그림의 한 구성요소에 대하여 서로 다른 위치와 방향을 가지게 할수 있다. 또는 그림의 개별적부분들을 꺼내고 후에 그 부분들을 화면에로 쉽게 다시 넣을수도 있다. 류사하게 모형화응용에서는 복잡한 물체 또는 체계의 부분들을 총체적인 계층관계에 따라 따로따로 만들어 놓고 위치를 정하게 할수 있다. 그리고 동화에서는 장면안에서 여러 물체들을 서로 다르게 움직이거나 부동으로 남아 있게 장면의 개별적인 부분들에 변환을 적용할수 있다.

1절. 구조물의 개념

PHIGS에서는 출력기초요소(및 관련속성)들의 표식이 있는 모임을 **구조물**(structure)이라고 부른다. 구조물과 같은 기초요소들의 표식이 있는 모임에 대하여 일반적으로 쓰는 다른 이름들로서는 토막(GKS에서)과 물체(Silicon Graphics체계의 Graphics Library에서)가 있다. 이 절에서는 PHIGS의 기본적인 구조물관리함수들을 고찰한다. 이러한 조작들은 그림의 기초요소들의 표식된 무리를 처리하는 다른 프로그람들에서도 사용할수 있다.

기본구조물함수

구조물을 만들 때 그 구조물에 대하여 지적되는 자리표위치와 속성값들은 중앙구조물기억기(central structure store)라고 하는 체계의 구조물목록에 표식이 있는 묶음으로 기억된다. 구조물은 함수

```
openStructure (id)
```

에 의하여 만든다. 구조물에 대한 표식은 파라메터 id에 할당되는 정의 옹근수이다. PHIGS+에서는 구조물을 표식하는데 옹근수대신에 문자렬이름을 리용할수도 있다. 이것은 구조물식별자를 기억하기 더 쉽게 한다. 모든 기초요소 및 속성들을 기입한후 구조물의 마지막끝은 closeStructure명령문으로 닫는다. 실례로 다음의 프로그람명령문들은 지적된 선형태와 색갈을 가지는 절선으로 된 선들의 순서렬로 구조물 6을 정의한다.

```
openStructure(6);
      SetLineType(lt);
      SetPolylineColourIndex(lc);
      Polyline(n, pts);
closeStructure;
```

하나의 그림에 대하여 임의의 개수의 구조물들을 만들수 있지만 한번에 하나의 구조물만 열수 있다(만들기처리에서). 임의의 열린 구조물은 새로운 구조물을 만들기전에 닫아야 한다. 이 요구는 closeStructure명령문에서 구조물식별번호가 필요없게 한다.

일단 구조물이 만들어 지면 그것은 선택된 출력장치에 함수

```
postStructure (ws, id, priority)
```

에 의해 현시할수 있다. 여기서 파라메터 ws는 워크스테이션식별자이며 id는 구조물의 이름, priority(우선권)에는 0~1사이의 실수값이 할당된다. 파라메터 priority에는 다른 구조물들에 대한 현시 우선권을 설정한다. 두개의 구조물이 하나의 출력표시장치에서 겹칠 때에는 더 높은 우선권을 가지는 구조물이 보이게 한다. 실례로 구조물 6과 9가 다음과 같은 우선권을 가지고 워크스테이션2에 보내면

```
postStructure(2, 6, 0.8)
postStructure(2, 9, 0.1)
```

구조물 6이 더 높은 우선권을 가지기때문에 구조물 6과 겹치는 구조물 9의 부분은 보이지 않게 된다. 두 구조물에 같은 우선권값이 할당되면 마지막에 보내진 구조물에 현시우선권이 주어 진다.

구조물이 능동워크스테이션에 보내지면 구조물안의 기초요소들을 주사하고 해석하여 선택된 출력장치(현시장치, 레이자인쇄기 등등)에 현시한다. 구조물목록에 대한 주사와 도형출력의 워크스테이션에로의 보내기를 송달(traversal)이라고 한다. 기초요소들에 대한 현재 속성값들의 목록은 송달상태목록이라고 하는 자료구조에 기억된다. 구조물보내기에서 변화가 일어 나면 체계의 구조물목록과 송달상태목록이 다같이 갱신된다. 이것은 자동적으로 워크스테이션에 보내진 구조물의 현시를 수정한다.

출력장치에서 구조물의 현시를 그만두게 하기 위하여서는 함수

```
unpostStructure(ws, id)
```

를 쓴다. 이것은 그 구조물을 지적된 출력장치에 대한 능동구조물목록에서는 지우지만 체계의 구조물목록에는 영향을 주지 않는다. 라스터체계에서 구조물은 기초요소들을 배경색으로 다시 그리는 방법으로 화면에서 제거한다. 그러나 이 처리는 지우려고 하는 구조물과 겹치는 다른 구조물들의 기초요소들의 현시에 영향을 미칠수 있다. 이것을 없애기 위하여서는 장면안의 여러 구조물들의 자리표범위를 리용하여 어느 구조물이 지우고 있는 구조물과 겹치는가를 결정할수 있다. 그러면 보내지 않을 구조물을 지운후 간단히 이 겹침구조물들을 다시 그리면 된다. 선택된 출력장치에서 모든 구조물들을 제거하는것은 함수

```
unpostAllStructures(ws)
```

에 의하여 실행할수 있다.

만일 개별적인 구조물을 체계의 구조물목록에서 지우려면 그것은 함수

```
deleteStructure(id)
```

에 의하여 수행한다. 물론 이것은 구조물을 받았던 모든 출력장치들에서도 현시를 그만 두게 한다. 구조물이 지워 진 다음에는 그 이름을 기초요소들의 다른 모임에서 다시 리용할수 있다. 체계의 구조물목록전체를 지워 버리는것은

```
deleteAllStructures
```

를 쓴다.
구조물을 다시 표식할수 있게 하는것도 편리하다. 이것은

```
changeStructureIdentifier(oldID, newID)
```

에 의하여 수행된다. 구조물의 표식을 변화시키는 한가지 리유는 여러 구조물들이 지워 진후 구조물들의 번호매기기를 고착정리하는것이다. 다른 하나는 구조물이 여러 위치에서 현시될 때 구조물의 위치선정을 검사하기 위하여 구조물표식들의 모임을 순환하기 위해서이다.

구조물속성의 설정

워크스테이션려파에 의해 구조물에 일정한 현시특성을 설정할수 있다. 려파에 의하여 설정할수 있는 세가지 특성은 보임성, 밝기강조, 대화식입력장치에 의한 구조물의 선택가능성이다.
선택된 매개 워크스테이션에서 구조물에 대한 보임성 및 비보임성설정은 함수

```
setInvisibilityFilter(ws, devCode, invisSet, visSet)
```

에 의하여 지적된다. 여기서 파라메터 invisSet에는 보이지 않게 할 구조물들의 이름이 들어 가며 파라메터 visSet에는 보이게 할 구조물들의 이름이 들어 간다. 비보임성려파로는 구조물들을 워크스테이션목록에서 실제로 지우지 않고 선택된 워크스테이션에서 구조물의 현시만을 켜기/끄기(on/off)절환할수 있다. 이것은 실례로 내부의 세부는 전혀 없이 건물의 륜곽선만을 보이게 하고 다음에 보임성을 재설정하여 모든 내부특징들을 가지고 건물을 볼수 있게 할수 있다. 지적할수 있는 추가적인 파라메터들로는 각 모임에 대한 구조물의 개수이다. 구조물은 라스터현시장치에서 구조물의 제거 및 지우기에 대하여 설명한것과 같은 절차를 리용하여도 보이지 않게 된다. 그러나 차이점은 구조물을 간단히 보이지 않게 하는 경우에는 장치에 대한 능동구조물목록에서 그것을 제거하지 않는다는것이다.
다른 편리한 구조물특성은 밝기강조이다. 지도화면에서 일정한 값 아래의 인구수를 가지는 모든 도시들을 더 밝게 할수도 있고 경치설계도에서 어떤 떨기나무들을 더 밝게 할수도 있으며 회로도에서 구체적인 전압범위안의 모든 요소들을 더 밝게 할수도 있다. 이것은 함수

```
setHighlightingFilter(ws, devCode, highlightSet, nohighlightSet)
```

에 의하여 진행한다. 파라메터highlightSet에는 밝기강조할 구조물들의 이름이 들어 가며 파라메터 nohighlightSet에는 밝기강조하지 않을 구조물들의 이름이 들어 간다. 강조구조물에 대하여 리용되는 밝기강조의 종류는 도형처리체계의 형과 능력에 관계된다. 색현시장치에서는 밝기강조구조물들을 더 밝은 세기 또는 밝기강조를 위하여 반전된 색으로 현시할수 있다. 다른 일반적인 밝기강조의 실현은 깜빡거리는 구조물이 현시되도록 보임성을 on과 off로 빨리 절환하는것이다. 깜빡거림은 또한 밝기강조되는 구조물의 세기를 낮은 값과 높은 값사이에서 빨리 교체하는 방법으로써도 실현할수 있다.
구조물에 대하여 설정할수 있는 세번째 현시특성은 잡기가능성이다. 이것은 구조물을 가리키거나 또는 그우에 화면지시자를 가져다 놓는것에 의해 선택되는 구조물의 능력을 말한다. 화면에서 어떤 구조물이 전혀 선택될수 없다는것을 담보하려 한다면 잡기가능성려파에서 그것을 잡기불가능이라고 선언할수 있다. 다음 장에서 입력방법들의 내용을 더 자세히 본다.

2절. 구조물의 편집

흔히 구조물을 만들고 닫은후 그것을 다시 수정하게 된다. 구조물의 수정은 설계응용에서 서로 다른 도형적인 배치들을 해 보거나 또는 새 시험자료에 의하여 설계구성을 변화시키는데 필요된다.

보충적인 기초요소들이 구조물에 추가하려면 이것은 간단히 openStructure함수에 의해 구조물을 다시 열고 요구되는 명령문들을 첨가하는 방법으로 진행할수 있다. 간단한 추가의 실례로서 다음의 프로그람토막에서는 먼저 채우기구역이 하나 있는 구조물을 만들고 후에 두번째 채우기구역을 구조물에 첨가한다.

```
openStructure (shape);
     setInteriorStyle (solid);
     setInteriorColourIndex (4);
     fillArea (nl, vertsl);
closeStrueture;
         .
         .
         .
openStructure (shape);
     setInteriorStyle (hollow) ;
     fillArea (n2, verts2);
closeStructure;
```

이 순서렬의 조작은 다음과 같이 초기부터 두개의 채우기구역을 가지는 구조물을 만드는것과 같다.

```
openStructure (shape);
     setInteriorStyle (solid);
     setInteriorColourlndex (4);
     fillArea (n1, verts1);
     setInteriorStyle (hollow) ;
     fillArea (n2, verts2);
closeStructure;
```

추가외에도 또한 때때로 구조물에서 어떤 항목을 지우거나 기초요소 또는 속성설정을 변화시키거나 구조물안의 선택된 위치에 묶음항목을 삽입할수 있다. 일반적인 편집조작은 구조물의 개별적인 구성요소에 대하여 순서번호를 호출하고 편집방식을 설정하는 방식으로 수행된다.

구조물목록과 요소지시기

출력기초요소 및 속성값과 같은 구조물안의 개별적인 항목들을 구조물요소 또는 간단히 요소라고 한다. 매개 요소에는 그것이 구조물안에 들어갈 때 참조위치값이 할당된다. 그림 7-1에서는 다음의 프로그람토막에 의하여 만들어 진 구조물요소들과 관련위치번호들의 기억상태를 보여 준다.

```
openStructure (gizmo) ;
      setLinetype (lt1);
      setPolylineColourIndex (lc1);
      polyline (n1, pts1);
      setLinetype (lt2) ;
      setPolylineColourIndex (lc2);
      polyline (n2, pts2);
closeStructure;
```

구조물요소들은 1부터 시작하여 옹근수값으로 련속적으로 번호 매겨 지며 값 0은 첫번째 요소앞의 위치를 가리킨다. 구조물이 열릴 때 요소지시기가 설정되고 구조물편집에 리용될수 있는 위치값이 할당된다. 열려 진 구조물이 새것이면(이미 체계의 구조물목록에 존재하는것이 아니면) 요소지시기는 0으로 설정된다. 열려 진 구조물이 이미 체계목록에 존재하는 것이라면 요소지시기는 그 구조물안의 마지막요소의 위치값으로 설정된다. 요소들이 구조물에 추가될 때마다 요소지시기는 1씩 증가된다.

그림 7-1. 구조물 gizmo에 대한 요소위치값

요소지시기의 값은 함수

```
setElementPointer(k)
```

에 의하여 구조물안의 임의의 위치에 설정할수 있다. 여기서 파라메터 k에는 0으로부터 구조물안의 최대요소수까지의 임의의 옹근수값을 할당할수 있다. 요소지시기의 위치를 지정하는데는 지시기를 현재 위치에 대하여 움직이게 하는 다음의 편위함수

```
offsetElementPointer(dk)
```

를 리용하여도 된다. 여기서 dk에는 지시기의 현재 위치로부터의 정 또는 부의 옹근수편위가 할당된다.

편집방식의 설정

구조물들은 두가지 가능한 방식들중 어느 하나로 수정할수 있다. 이것을 구조물의 편집방식이라고 한다. 편집방식의 값은

```
setEditMode(mode)
```

에 의하여 설정한다. 여기서 파라메터 mode에는 값 insert(삽입) 또는 replace(교체)가 할당된다.

구조물요소의 삽입

편집방식이 insert로 설정되였을 때 구조물에 들어 가는 항목은 요소지시기의 그다음의 위치에 놓여 지게 된다. 이때 구조물목록안에서 삽입된 항목 이후의 요소들은 번호가 자동적으로 다시 매겨진다.

삽입조작을 설명하기 위하여 구조물gizmo에서 현재의 표준선너비를 약간 다른 값으로 변화시키자(그림7-2). 이것은 선너비명령문을 첫번째 polyline명령문앞의 임의의 곳에 삽입하면 될수 있다.

```
openStructure (gizmo);
      setEdibMode (insert);
      setElementPointer (0);
      setLinewidth (lw);
           .
           .
           .
closeStructure;
```

그림 7-2에서는 이 삽입조작에 의하여 만들어 진 gizmo의 수정된 요소목록을 보여 주었다. 이 삽입후 요소지시기에는 값 1(새로운 선너비속성의 위치)이 할당된다. 또한 선너비명령문 이후의 모든 요소들은 값 2에서 시작하여 다시 번호 매겨 졌다.

그림 7-2. 구조물 gizmo에 선너비 속성을 삽입한후 수정된 요소목록과 요소지시기의 위치

새로운 구조물이 만들어 지는 경우에는 편집방식이 자동적으로 값 insert로 설정된다. 구조물을 다시 열기전에 편집방식이 이 기정값으로부터 변화되지 않았다고 하면 제7장 제2절의 시작에서 보여 준바와 같이 편집방식이나 또는 요소지시기에 대한 값설정이 없이 요소목록의 마지막에 항목들을 추가할수 있다. 이것은 다시 열려 지는 구조물에서는 편집방식이 값 insert로 되여 있고 요소지시기는 목록의 마지막요소를 가리키기때문이다.

구조물요소의 교체

편집방식이 값 replace로 설정되는 경우에 구조물에 들어 가는 항목은 요소지시기의 위치에 놓여진다. 그 위치의 본래 요소는 지워 지고 요소지시기의 값은 변화되지 않는다.

교체조작의 실례로서 구조물 gizmo(그림 7-1)에서 두번째 polyline의 색을 변화시키려면 순서렬

```
openStructure (gizmo);
      setEditMode (replace) ;
      setElementPointer (5);
      setPolylineColourIndex (lc2New) ;
           .
           .
           .
closeStructure;
```

에 의하여 수행할수 있다. 그림 7-3은 두번째 polyline에 대하여 새로운 색갈을 준 gizmo의 요소목록을 보여 준다. 바꾸어넣기조작후 요소지시기는 위치 5(새로운 선 색갈속성위치)에 남는다.

그림 7-3. 구조물 gizmo에서 두번째 polyline의 색갈을 변화시킨후 수정된 요소목록과 요소지시기의 위치

구조물요소의 지우기

요소지시기의 현재 위치의 요소는 함수

```
deleteElement
```

에 의하여 지워 버릴수 있다. 이것은 구조물에서 요소를 제거해 버리며 이때에 요소지시기의 값은 그 앞의 요소에 설정된다.

요소지우기 실례로서 구조물 gizmo(그림 7-1)에서 동일한 색으로 현시되는 두개의 polyline을 쓰기로 결심했다고 하자 . 이것은 두번째 색갈속성을 지워 버리면 될수 있다.

```
openStructure(gizmo);
      setElementPointer(5);
      deleteElement;
           .
           .
           .
closeStructure;
```

요소지시기는 다음에 값 4로 재설정되며 뒤따르는 모든 요소들은 그림 7-4에 보여 준바와 같이 번호가 다시 매겨 진다.

그림 7-4. 구조물 gizmo에서 두번째 polyline에 대한 색갈속성명령문을 지운후 수정된 요소목록과 요소지시기의 위치

구조물요소들의 련속된 무리는 함수

```
deleteElementRange(k1, k2)
```

에 의하여 지워 버릴수 있다. 여기서 옹근수파라메터 k1은 시작위치번호를 주며 k2는 마지막위치번호를 지적한다. 실례로 구조물 gizmo에서 두번째 polyline과 관련속성들은

```
deleteElementRange(4, 6)
```

에 의하여 지워 버릴수 있다. 그리고 구조물안의 모든 요소들은 함수

```
emptyStructure(id)
```

에 의하여 지워 버릴수 있다.

구조물요소의 표식붙이기

구조물에 대하여 많은 수정을 하면 후에 요소위치들의 자리길을 잃어 버리기 쉽다. 요소들을 지우거나 삽입하면 요소위치들의 번호가 달라 진다. 함수

```
label(k)
```

에 의하여 구조물안의 서로 다른 요소들에 간단히 표식을 붙이면 수정을 할 때 새로운 위치번호의 자리길을 보존하는것을 피할수 있다. 여기서 파라메터 k는 옹근수위치식별자이다. 표식은 위치번호의 참조없이 구조물요소들의 위치를 선정하는것을 돕는것으로서 구조물목록안의 임의의 곳에 삽입될수 있다. 표식함수는 구조물의 송달처리에는 영향을 주지 않으면서 구조물요소를 만든다. 구조물에 기억된 표식은 개별적인 요소위치의 리용이 아니라 단순히 편집참조로 리용된다. 또한 구조물요소의 표식붙이기는 유일하지 않아도 된다. 때때로 특히 동일한 편집조작이 구조물의 여러 위치에 적용되는 경우에는 둘 또는 그이상의 요소에 동일한 표식값을 주면 편리하다.

표식붙이기의 리용을 설명하기 위하여 그림 7-5에 보여 준것과 같은 요소들과 위치번호를 가지는 구조물 labeledGizmo를 다음의 루틴으로 만든다.

```
openStructure (labeledGizmo);
          label (objectlLinetype) ;
          setLinetype (lt1);
          label (objectlColor) ;
          setPolylineColourIndex (lc1);
          label (object1) ;
          polyline (n1, pts1);
          label (object2Linetype) ;
          setLinetype (lt2);
          label (object2Color) ;
          setPolylineColourIndex (lc2);
          label (object2) ;
          polyline (n2, pts2);
closeStructure;
```

이제 이 구조물의 임의의 기초요소 또는 속성을 변화시키려 한다면 표식을 참조하는것에 의하여 그것을 할수 있다. 비록 이 구조물에서는 모든 항목들에 표식을 붙였지만 예견되는 편집의 형과 수량에 따라 다른 표식붙이기방법들도 리용될수 있다. 실례로 모든 속성들을 하나의 표식아래에서 총괄할수도 있으며 모든 색갈속성들에는 동일한 표식식별자를 줄수도 있다.

표식은 함수

```
setElementPointerAtLabel (k)
```

에 의하여 참조되는데 이 함수는 요소지시기를 파라메터 k의 값으로 설정한다. 표식에 대한 조사는 현재의 요소지시기 위치에서 시작하여 요소목록을 따라 앞으로 전진해 나간다. 이것은 구조물을 다시 열 때 지시기를 재설정하여야 된다는것을 의미한다. 왜냐하면 다시 열려 진 구조물에서는 지시기가 항상 마지막요소를 가리키는데 표식조사는 요소목록을 따라 뒤로는 진행되지 않기때문이다. 실례로 구조물 labeledGizmo에서 두번째 물체의 색을 변화시키려면 적당한 색갈속성명령문표식을 조사하기 위하여 구조물을 다시 연후 지시기를 시작위치에 다시 설정한다.

	0 구조물 labeledGizmo
1	label (objectlColor)
2	setLinetype (lt1)
3	label (object1Colour)
4	setPolylineColourIndex (lc1)
5	label (object1)
6	polyline (n1, pts1)
7	label (object2Linetype)
8	setLinetype (lt2)
9	label (object2Color)
10	setPolylineColourIndex (lc2)
11	label (object2)
요소 지시기 → 12	polyline (n2, pts2)

그림 7-5. 구조물 labeledGizmo에 기억된 표식 붙은 물체들과 관련위치번호들의 모임

```
openStructure (labeledGizmo);
          setElementPointer (0);
          setEditMode (replace);
          sefcElementPointerAtLabel (object2Color);
          offsetElementPointer (1);
          setPolylineColourIndex (lc2New);
                 .
                 .
                 .
closeStrueture;
```

표식에 의하여 참조되는 항목지우기는 마지막openStructure루틴에서 설명된 바꾸어넣기조작과 류사하다. 먼저 해당한 표식의 위치를 선정하고 다음에 지시기를 편위시킨다. 실례로 구조물 labeledGizmo에서 두번째 polyline에 대한 색갈속성은 순서렬

```
openStructure (labeledGizmo);
          setElementPointer (0);
          setEditMode (replace);
          setElementPointerAtLabel (object2Color) ;
          offsetElementPointer (1);
          deleteElement ;
                 .
                 .
                 .
closeStructure;
```

에 의하여 지워 버릴수 있다. 또한 지적된 표식들사이의 구조물요소무리는 함수

```
deleteElementsBetweenlabels(k1, k2)
```

에 의하여 지워 버릴수 있다. 요소들의 모임이 지워 진후 요소지시기는 위치 k1로 설정된다.

한 구조물로부터 다른것에로 요소들의 복사

지적된 구조물의 모든 항목들을 열려 진 구조물에 복사하는것은

```
copyAllElementsFromStructure(id)
```

에 의하여 수행할수 있다. 구조물 id로부터 요소들은 열려 진 구조물안에 요소지시기의 다음 위치에서 시작하여 편집방식의 설정에는 무관계하게 삽입된다. 복사조작이 수행되면 요소지시기는 열려 진 구조물에서 삽입된 마지막항목의 위치에 설정된다.

3절. 모형화기초개념

구조물의 중요한 리용은 여러가지 형태의 체계들의 설계 및 표현에 있다. 건물설계와 전자회로와 같은 건축 및 공학체계들에서는 일반적으로 다 콤퓨터지원설계방법을 리용한다. 도형적인 방법들은 또한 경제, 재정, 조직, 과학, 사회, 환경체계의 표현에도 리용된다. 흔히 체계들에 대한 표현은 여러가지 조건하에서 체계의 움직임을 모의하기 위하여 만들어 진다. 모의결과는 교육수단으로나 또는 체계에 대한 어떤 결심채택의 기초로 될수 있다. 이런 응용들에서 여러가지 효과를 내자면 도형처리 프로그람들이 체계에 대한 도형적인 표현을 만들고 조작하는 효과적인 방법들을 가지고 있어야 한다.

체계에 대한 표현을 만들고 조작하는것을 **모형화**라고 한다. 임의의 하나의 표현을 체계의 **모형**이라고 한다. 체계에 대한 모형들은 도형적으로 정의될수도 있고 체계파라메터들사이의 관계를 정의하는 식들의 모임과 같이 순수 서술적인것일수도 있다. 도형적인 모형을 흔히 **기하학적모형**이라고 한다. 왜냐하면 체계의 구성부분들이 선, 다각형, 원과 같은 기하학적인 실체들로 표현되기때문이다. 여기서는 모형이라는 용어를 도형처리적응용만을 상대하여 콤퓨터로 발생시킨 체계에 대한 기하학적인 표현의 의미로 사용하겠다.

모형의 표현

그림 7-6은 많은 체계모형들에서의 공통인 특징을 설명해 주는 론리회로에 대한 표현을 보여 준다. 체계의 구성부분들은 **기호**라고 부르는 기하학적인 구조물로 현시되며 기호들사이의 관계는 이 실례에서 련결선들의 망으로 표현한다. 론리연산 and, or, not에 대한 론리문을 표현하는데 3개의 표준적인 기호가 리용되고 있다. 련결선들은 체계부분들사이의 입출력흐름(왼쪽에서 오른쪽에로)에 의하여 관계를 정의한다. 하나의 기호 and문은 론리회로안에서 두개의 서로 다른 위치에서 현시되고 있다. 몇개의 기본기호들을 반복적으로 배치하는것은 복잡한 모형을 만드는 일반적인 방법이다. 모형안에서 매 기호의 이런 출현을 그 기호의 구체례라고 한다. 그림 7-6에는 or 및 not기호에 대하여서는 1개의 구체례, and기호에 대하여서는 2개의 구체례가 있다.

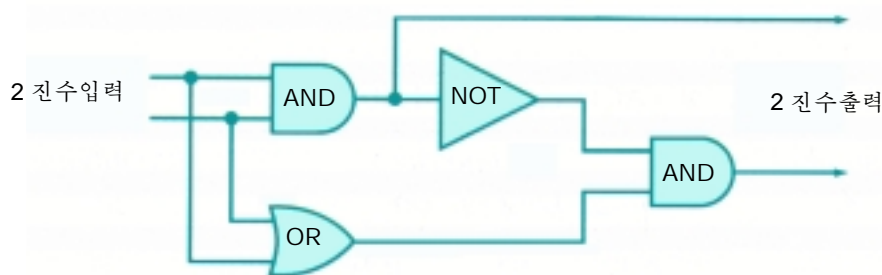



그림 7-6. 론리회로의 모형

많은 경우들에 체계의 부분들을 표현하기 위하여 선택되는 매개 도형적인 기호들은 체계의 서술에 의하여 지시된다. 회로모형에서는 표준적인 전기 또는 론리기호들이 리용된다. 정치, 재정, 경제체계와 같은 추상적인 개념들을 표현하는 모형들에서의 기호들은 임의의 편리한 기하학적인 모양이 될수 있다.

모형을 표현하는 정보들은 보통 기하학적, 비기하학적자료들의 결합으로 제공된다. 기하학적정보에는 구성부분들의 위치를 규정하는 자리표위치, 부분들의 구조물을 정의하는 출력기초요소들과 속성

함수, 부분들사이를 련결시키는 자료들이 포함된다. 비기하학적정보에는 본문표식, 모형의 동작특성을 서술하는 알고리듬, 구성부분들사이의 관계 또는 련결을 결정하는 규칙(기하학적자료로 지적되지 않으면)들이 포함된다.

모형을 만들고 조작하는데 필요한 정보를 지적하는데는 두가지 방법이 있다. 한가지 방법은 정보를 표 또는 련결목록과 같은 자료구조로 기억시키는것이다. 다른 방법은 정보를 절차에서 지적하는것이다. 일반적으로 모형표현은 자료구조와 절차를 다같이 포함하게 된다. 하지만 일부 모형들은 완전히 자료구조로만 정의되거나 다른것들은 절차지적만을 리용하기도 한다. 물체의 고체모형화를 진행하는 응용에서는 주로 자리표위치를 정의하는 일부 자료구조로부터 취해 지는 정보와 대단히 적은 절차들을 리용하게 된다. 한편 기상학적모형은 주로 온도와 압력의 변화를 계산하는 절차들이 필요하게 된다.

자료구조와 절차의 결합이 어떻게 쓰이는가 하는 실례로서 그림 7-6의 론리회로에 대한 몇가지 모형지적을 고찰하자. 한가지 방법은 론리요소들을 망련결이 어떻게 되여 있고 회로가 어떻게 동작하는가를 지적하는데 리용되는 처리절차들과 함께 자료표(표 7-1)로 정의하는것이다. 이 표에서 기하학적자료에는 자리표와 문을 그리며 위치를 정하는데 필요한 파라메터들이 포함된다. 이 기호들은 모두 다각형형태로 그려 지거나 선분과 타원호의 결합으로 형성될수 있다. 매개 구성요소들에 대한 표식도 표에 포함되지만 기호들이 일반적으로 인정되는 모양으로 현시된다면 표식들은 빼버릴수 있다. 한편 절차는 문들의 자리표위치와 그것들의 련결에 대한 순서지적에 기초하여 문을 현시하고 련결선을 만드는데 리용될수 있다. 임의로 주어 지는 입력에 대한 회로출력(2진값)을 만들기 위하여 추가적인 절차가 리용된다. 이 절차는 마지막출력만을 현시하도록 설정될수도 있고 회로의 내부기능을 설명하기 위한 중간출력값을 현시하도록 설계될수도 있다.

표 7-1. 그림 7-6의 회로에서 매개 문의 구조와 위치를 정의하는 자료표

기호코드	기하학적표현	식별표식
문 1	(자리표와 기타 파라메터)	AND
문 2	:	OR
문 3	:	NOT
문 4	:	AND

또 다르게 회로모형에 대한 도형적인 정보를 자료구조로 지적할수 있다. 그러면 문뿐아니라 련결선들도 회로에서 그 끝점들을 명백히 기입한 자료표로 정의할수 있다. 그다음 하나의 절차가 회로를 현시하고 출력을 계산하게 된다. 다른 극단으로 외부적인 자료구조를 리용함이 없이 절차들로만 모형을 모두 정의할수도 있다.

기호계층

많은 모형들은 기호들의 계층구조로 조직할수 있다. 모형에 대한 기초적인 《블로크》를 고찰하는 모형의 형에 적합한 간단한 기하학적형태들로 정의한다. 이 기초기호들은 **모듈**(module)이라고 부르는 복합물체를 형성하는데 리용할수 있다. 또 모듈자체는 하나로 묶어 져 더 높은 준위의 모듈을 형성한다. 이런 식으로 다시 묶어 나가면 모형의 여러가지 구성부분들을 형성할수 있다. 제일 간단한 경우 모형을 그림 7-7에서와 같이 구성부분들의 1준위계층으로 표현할수 있다. 이 회로실례에서는 문들이 매개 문에 대한 서술에서 지적되는 련결규칙에 따라 위치가 정해 지고 서로 직선으로 련결된다

고 가정한다. 이 계층적인 서술에서 기초기호는 론리문이다. 비록 문자체를 직선, 타원호, 본문들로 형성되는 계층으로 서술할수도 있지만 이런 서술은 론리회로를 만드는데 적합하지 못하다. 여기서 제일 간단한 블로크는 문이다. 서로 다른 기하학적형태설계에 흥미를 가지는 응용에서는 기초기호가 선분과 호로 정의될수 있다.

2준위기호계층의 실례를 그림 7-8에 보여 주었다. 여기서 사무실설계에서는 작업구역들의 배치를 진행한다. 매개 작업구역에는 해당한 비품들이 갖추어 진다. 기초기호는 비품항목들 즉 작업대, 의자, 당반, 서류장 등이다. 더 높은 준위의 물체는 작업구역으로서 그것은 서로 다른 비품구성으로 형성된다. 기초기호들의 구체례는 매개 작업구역안에서 그것의 크기, 위치, 방향을 지적하여 정의한다. 물체들의 크기가 고정된 사무실설계프로그람에서는 사용자가 위치와 방향만을 지적하면 된다. 위치는 작업구역에서의 자리표위치로 주어 지며 방향은 기호가 어느쪽으로 향하는가를 결정하는 회전으로 지적된다. 계층의 두번째 준위에서 매개 작업구역은 그의 크기, 사무실안에서의 위치와 방향를 지적하는 방법으로 정의한다. 매개 작업구역에 대한 경계는 작업구역을 둘러 싸며 사무실안에서 통로를 제공하는 칸막이에 의하여 맞추어 지게 된다.

보다 복잡한 기호계층은 더 높은 매 준위에서 기호집단들을 반복적으로 묶어 주는 방법으로 만든다. 그림 7-8의 사무실설계는 서로 다른 방, 건물의 서로 다른 층, 건물집단안에서의 서로 다른 건물, 널리 분리된 물리적인 위치에서의 서로 다른 지역복합체를 만드는 기호집단이 포함되도록 확장할 수 있다.

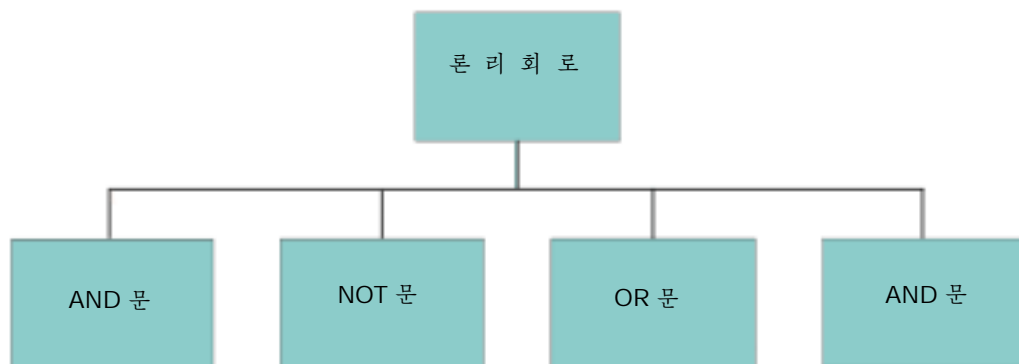



그림 7-7. 론리문으로 형성되는 회로의 1준위계층서술

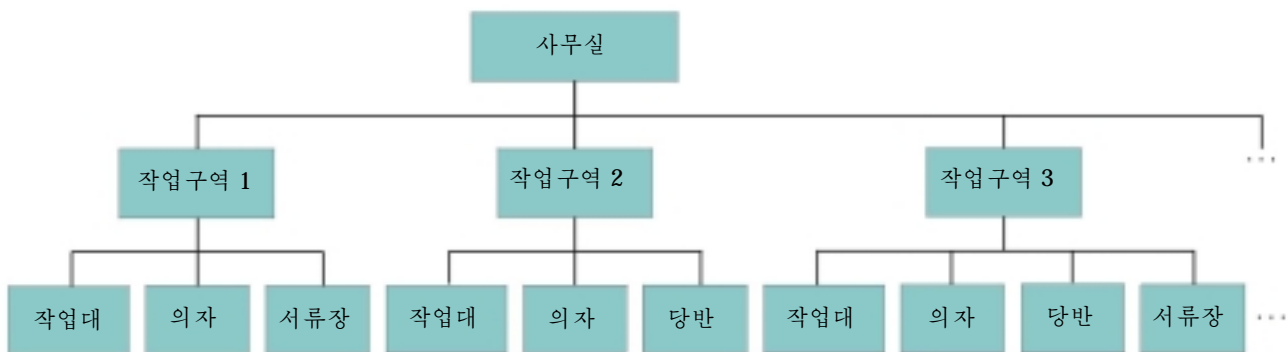



그림 7-8. 사무실설계의 2준위의 계층서술

모형화프로그람

일부 일반목적도형처리체계 실례로 GKS는 여러 방면의 모형화목적에 순응하도록 설계되여 있지 않다. 모형화절차 및 자료구조를 취급하는데 필요한 루틴들은 흔히 개별적인 다른 모형화프로그람으로 설정되며 그러면 도형처리프로그람들은 모형화프로그람과 대면부를 맞추게 된다. 도형처리루틴의 목적은 마지막출력화면을 만들고 조작하는 방법을 제공하는것이다. 반대로 모형화루틴은 기호계층에 의하여 모형표현들을 정의하고 재정렬하는 방법을 제공한다. 그것들은 그다음 도형처리루틴에 의하여 처리되고 현시된다. PHIGS와 SiliconGraphics equipment의 Graphics Library(GL)와 같은 체계들은 모형화 및 도형처리함수들을 하나의 프로그람으로 통합시키도록 설계되여 있다.

응용모형화프로그람에서 사용가능한 기호들은 그 프로그람에서 취급하도록 계획한 응용목적의 형태에 따라 정의되고 구조화된다. 모형화프로그람은 2차원 또는 3차원화면에 대하여 설계될수 있다. 그림 7-9에서는 회로설계에 리용된 2차원설계도면을 보여 주었다. 3차원분자모형화의 실례를 그림 7-10에 보여 주고 3차원사무실설계도면을 그림 7-11에 주었다. 이런 3차원화면들은 설계자에게 설계도면의 모양에 대한 리해를 더 좋게 해 준다. 다음 절에서는 모형화프로그람들의 특징, 모형화함수들과 도형처리루틴들의 결합이나 통합에 대한 방법들을 찾아 본다.

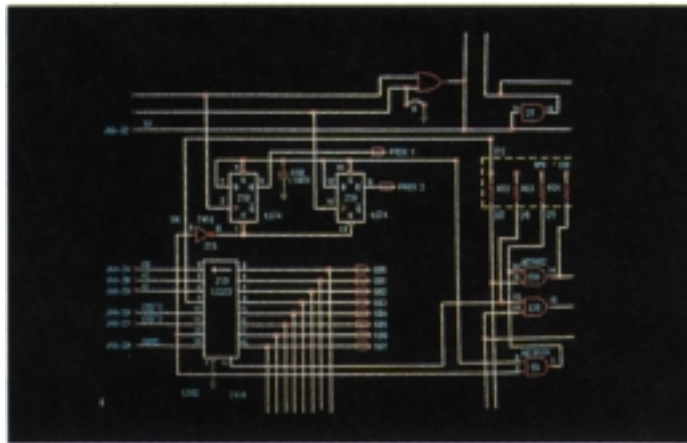



그림 7-9. 회로설계에 리용된 2차원모형화설계도면

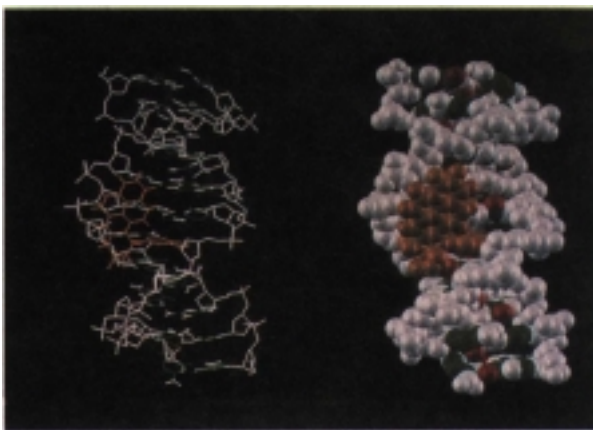



그림 7-10. DNA의 3차원분자모형을 보여 주는 립체화상쌍의 절반

그림 7-11. 사무실설계도면의 3차원상

4절. 구조물에 의한 계층적인 모형화

체계의 계층적인 모형은 구조물들을 나무구조를 형성하도록 종속련결시킨 구조로 만들수 있다. 매개 구조물이 계층에 놓여 질 때에는 모형전체에 알맞게 맞추어 지도록 하는 적당한 변환이 할당되게 된다. 사무기관의 사무실설계를 상상해 보면 여기에서는 비품들이 사무방들과 작업구역들에 배치되고 그다음 하나의 부서에 사무방들과 작업구역들이 배치된다. 이런 식으로 더 높은 계층으로 갈수 있다.

국부자리표와 모형화변환

많은 설계응용들에서 모형들은 기초기호모임으로 정의되는 기하학적인 형태들의 구체례(변환된 복사물)들로 만들어 진다. 구체례들은 모형의 세계자리표계안에서 기초기호들의 위치를 지정하여 만든다. 응용에 리용될 여러가지 도형적인 기호들은 모형화자리표계라고 하는 독립적인 자리표계에서 각각 정의된다. 모형화자리표는 또한 국부자리표 또는 때때로 주자리표라고도 한다. 그림 7-12에는 2차원사무실설계에 리용되는 두개의 기호에 대한 국부자리표정의를 보여 주었다.

도형적인 모형의 구성부분을 만들기 위하여 세계자리표에서 기호의 구체례가 얻어 지도록 기호의 국부자리표정의에 변환을 적용하게 된다. 기호의 모형화자리표정의에 적용되는 변환을 모형화변환이라고 한다. 일반적으로 모형화변환에는 세계자리표로 기호의 위치를 지정하는 이행, 회전, 비례변환이 포함되는데 일부 응용들에서는 다른 변환들도 리용될수 있다.

의자와 자리표배렬

x 의자	y 의자
-3	-3
-3	3
-3	3
3	-3
1	3
2	0
1	3

ㄱ) 의자

작업대자리표배렬

x 작업대	y 작업대
0	0
-8	0
-8	6
8	6
8	-10
0	-10

ㄴ) 작업대

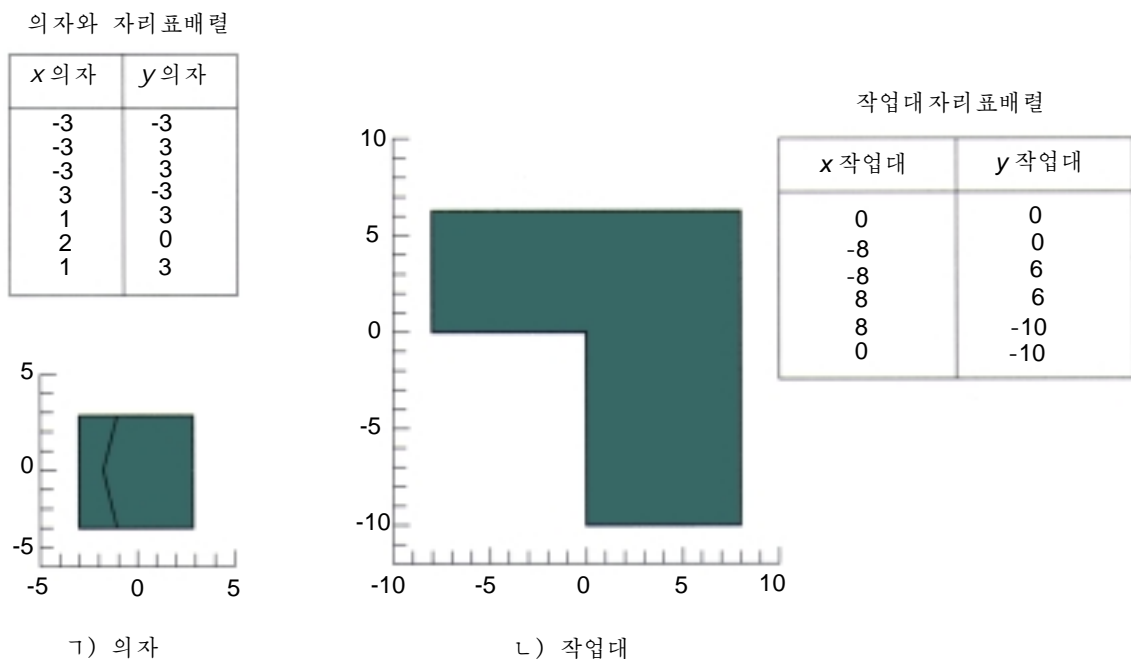



그림 7-12. 국부자리표에서 정의되는 물체

모형화변환

5장에서 설명한 기하학적인 변환함수들을 리용하여 매개 모형화변환행렬을 얻는다. 즉 모형화변환을 수행하도록 개별적인 변환행렬들을 설정할수도 있고 변환파라메터들을 입력하고 체계가 행렬들을 만들게 할수도 있다. 어느 경우나 모형화프로그람들은 개별적인 변환들을 련결하여 동차자리표모형화변환행렬 **MT**를 만든다. 세계자리표에서 기호의 구체례는 다음에 모형화자리표위치 (P_{mc})에 **MT**를 적용하여 대응하는 세계자리표위치 ($P_{\omega c}$)를 구함으로써 얻어 진다.

$$\mathbf{P}_{wc} = \mathbf{MT} \cdot \mathbf{P}_{mc} \qquad (7-1)$$

구조물계층

앞에서 본바와 같이 모형화응용에서는 일반적으로 기초기호들이 모듈이라고 부르는 무리로 합성되고 이 모듈들은 더 높은 준위모듈로 결합되고 또 그다음 같은 식으로 진행할수 있다. 이러한 기호계층은 나무의 매개 련속된 준위에 있는 구조물안에 구조물을 포함시키는 방법으로 만들수 있다. 처음에 모듈(구조물)을 기호구체례들과 그것들의 변환파라메터들의 목록으로 정의할수 있다. 다음 준위들에서는 더 낮은 모듈의 구체례들과 그것들의 변환파라메터들의 목록으로 더 높은 준위의 모듈을 정의한다. 이런 처리는 나무의 뿌리까지 계속되는데 뿌리는 세계자리표에서의 전체 그림을 표현한다.

구조물을 다른 구조물안에 넣는것은 함수

```
executeStructure(id)
```

에 의하여 수행한다.

구조물에 적당한 방향을 주기 위하여서는 먼저 구조물 id에 해당한 국부변환을 할당한다. 이것은

```
setLocalTransformation(mlt, type)
```

에 의하여 수행된다. 여기서 파라메터 mlt는 변환행렬을 가리킨다. 파라메터 type는 현재의 모형화변환행렬에 대하여 수행할 행렬합성의 형을 지적하는데 세가지 값 pre, post, replace중 어느 하나가 할당된다. 현재의 변환행렬을 mlt로 단순히 교체하려면 파라메터 type를 값 replace로 설정한다. 현재행렬에 이 함수에서 지적하는 국부행렬을 왼쪽곱하기하려고 하면 pre를 선택한다. 값 post에 대해서도 류사하다. 다음의 코드토막은 물체의 첫번째 구체례를 뿌리마디아래의 계층에 설정하는 모형화명령문들의 순서렬을 보여 준다.

```
createStructure (id0);
      setLocalTransformation (Imt, type);
      executeStructure (id1);
           .
           .
           .
closeStructure;
```

동일한 절차를 구조물 id0안에서 다른 물체들의 구체례를 주는데도 리용하여 계층의 이 준위에 다른 마디들을 설정한다. 그다음 구조물 id0안에 있는 id1과 다른 구조물들안에서 다시 물체들의 구체례를 줌으로써 나무의 다음 아래준위를 만들수 있다. 이 처리를 나무가 완성될 때까지 반복한다. 전

체 나무는 뿌리마디 즉 앞의 실례에서는 구조물 id0을 보내여 현시한다. 다음의 프로그람에서는 계층적인 구조물이 물체를 모형화하는데 어떻게 리용될수 있는가를 보여 준다.

```
void main ()
{
 enum { Frame, Wheel, Bicycle };
 int nPts;
 wcPt2 pts[256] ;
 pMatrix3 m;
 /* Routines to generate geometry */
 extern void getWheelVertices (int * nPts, wcPt2 pts);
 extern void getFrameVertices (int * nPts, wcPt2 pts);

 /* Make the wheel structure */
 getWheelVertices (nPts, pts);
 openStructure (Wheel);
 setLineWidth (2. 0) ;
 polyline (nPts, pts);
 closeStructure;
 /* Make the frame structure */
 getFrameVertices (nPts, pts);
 openStructure (Frame) ;
 setLineWidth (2. 0);
 polyline (nPts, pts);
 closeStructure ;

 /* Make the bicycle */
 openStructure (Bicycle) ;
 /* Include the frame */
 executeStructure (Frame) ;
 /* Position and include rear wheel */
 matrixSetIdentity (m) ;
 m[0,2] := -1. 0; m[1,2] := -0. 5;
 setLocalTransformationMatrix (m, REPLACE) ;
 executeStructure (Wheel);
 /* Position and include front wheel */
 m[0,2] := 1. 0; m[1,2] := -0. 5;
 setLocalTransformationmatrix (m, REPLACE) ;
 executeStructure (Wheel);
 closeStructure;
}
```

계층은 함수

```
deleteStructureNetwork (id)
```

에 의해 지운다. 여기서 파라메터 id는 나무의 뿌리구조물을 가리킨다. 이 함수는 계층이 나무로 조직되였다고 가정하고 계층의 뿌리마디와 executeStructure함수를 리용하여 뿌리아래에 넣어 진 모든 구조물들을 지워 버린다.

요약

구조물(일부 체계들에서는 토막 또는 대상이라고 한다.)은 출력기초요소들과 련관된 속성들의 표식이 있는 묶음을 말한다. 구조물들의 모임으로 만들어 진 그림에서는 그림의 구성요소들을 서로 독립적으로 쉽게 추가, 삭제, 관리할수 있다. 구조물들이 만들어 지면 중심구조물기억기에 기억되게 된다. 그다음 구조물들을 할당된 우선권을 가지고 여러가지 출력장치들에 보내면 현시되게 된다. 두 구조물이 겹칠 때에는 보다 높은 우선권을 가지는 구조물이 보다 낮은 우선권을 가지는 구조물우에서 현시된다.

구조물에 대하여 보임성, 밝기와 같은 속성들을 설정하는데는 워크스테이션려파를 사용할수 있다. 구조물은 보임성려파로서 구조물목록에 그것이 있는 한 구조물의 현시를 그만둘수 있다. 밝기강조려파는 깜박거림, 색, 더 밝은 세기를 가지고 물체를 강조하여 현시하는데 쓴다.

구조물에는 여러가지 편집조작을 적용할수 있다. 구조물을 다시 열고 추가, 삽입,지워 버리기조작들을 수행할수 있다. 구조물의 위치를 요소지시기라고 한다. 보충적으로 구조물안의 기초요소들이나 속성들에 개별적으로 표식을 붙일수 있다.

도형처리응용에서 모형이라는 용어는 어떤 체계에 대한 도형적인 표현을 가리킨다. 체계들에서 구성부분들은 국부(모형화)자리표계로 정의되는 기호들로써 표현한다. 전기회로에서와 같이 많은 모형들은 기호들의 구체례를 지정된 위치에 배치하여 만든다.

모형들은 기호들의 계층으로 만들어 진다. 실례로 자전거는 자전거틀과 바퀴로 만들어 지게 된다. 틀에는 손잡이, 발디디개와 같은 부분들이 들어 간다. 그리고 바퀴는 살, 테, 다이야를 포함한다. 계층적인 모형은 종속련결된 구조물들에 의하여 만들수 있다. 실례로 틀구조물과 바퀴구조물을 포함하는 자전거구조물을 설정할수 있다. 틀과 바퀴구조물들은 다같이 기초요소들과 보충적인 구조물들을 포함할수 있다. 구조물종속련결은 오직 하나의 기초요소(혹은 속성)들만 포함하는 구조물까지 계속해 내려 간다.

매개 구조물이 다른 구조물안에 들어 갈 때 해당한 모형화변환이 종속되는 구조물에 적용될수 있다. 이 변환은 구조물을 계층속에 맞추어 넣기 위하여 필요한 방향설정과 비례를 서술한다.

참고문헌

PHIGS에서의 구조물조작과 계층모형화는 Hopgood와 Duce(1991), Howard들(1991), Gaskins(1992), Blake(1993)에서 설명하였다.

GKS에서의 토막조작에 대한 정보는 Hopgood(1983)과 Enderle(1984)들을 보시오.

련습문제

7-1. 중심구조물기억기에 정보를 만들고 조작하는 프로그람을 쓰시오. 이 프로그람은 open Structure, deleteStructure, changeStructureIdentifier 와 같은 함수들에서 호출되게 된다.

7-2. traversal상태목록에 정보를 기억시키는 루틴을 쓰시오.

7-3. 화면에서 현시된 모든 구조물에 대하여 자리표범위가 주어 졌을 때 라스터체계에서 지적된 구조물을 지워 버리는 루틴을 쓰시오.

7-4. 라스터체계에서 unpostStructure 함수를 실현하는 프로그람을 쓰시오.

7-5. 라스터체계에서 deleteStructure 함수를 실현하는 프로그람을 쓰시오.

7-6. 깜박거림동작으로 밝기강조를 실현하는 프로그람을 쓰시오.

7-7. 구조물을 편집하기 위한 루틴들을 쓰시오. 이 루틴들은 구조물요소들의 추가, 삽입, 교체, 지우기와 같은 형태의 편집조작을 주어야 한다.

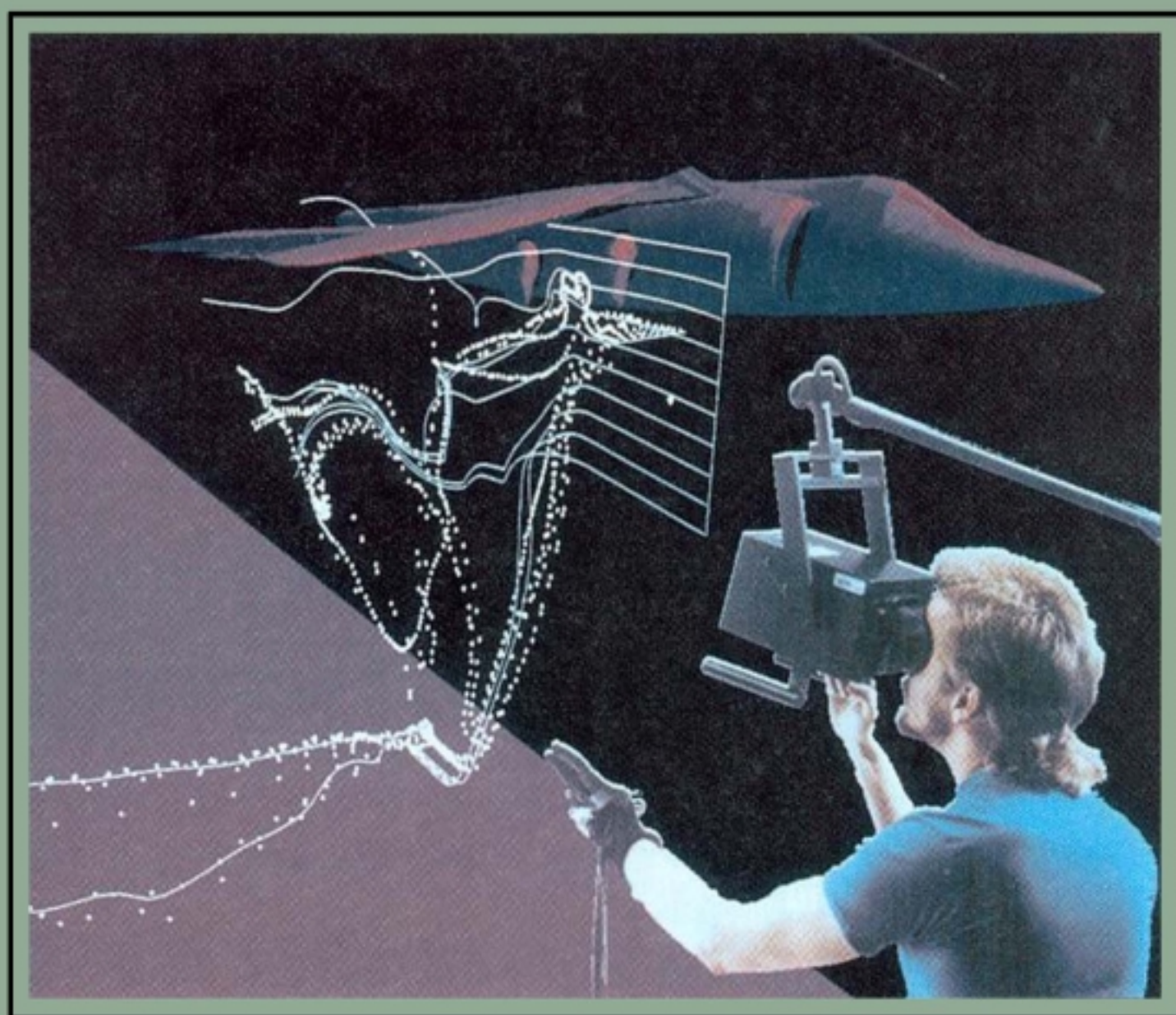
7-8. 명백히 차이나는 몇개 체계에 대하여 알맞는 모형표현을 고찰하시오. 매 체계에 대하여 도형적인 표현들을 어떻게 실현하겠는가에 대해서도 고찰하시오.

7-9. 그림 7-6과 같은 론리회로모형화응용에 대하여 회로를 현시하는데 쓰일 표준론리기호들의 구체적인 도형을 서술하시오.

7-10. 회로에서 사용자가 전기기호들의 위치를 지적할수 있는 전기회로모형화프로그람을 개발하시오. 차림표안에서 전기기호의 구체례를 회로에 놓는데는 오직 이동만 필요하다. 일단 구성요소들이 회로에 배치되면 그것들은 지적된 다른 구성요소들과 직선으로 련결되게 된다.

7-11. 2차원사무실설계프로그람을 창안하시오. 비품형태의 차림표가 설계자에게 제공되여야 한다. 설계자는 하나의 방안(1준위계층)에서 물체를 임의의 위치에 놓을수 있다. 구체례변환은 이동과 회전으로 제한한다.

7-12. 비품형태에 대한 차림표를 제시하는 2차원사무실설계프로그람을 창안하시오. 2준위계층이 사용되게 되는데 비품항목들이 여러 작업구역들에 배치될수 있고 작업구역들은 더 큰 구역으로 배치된다. 구체례변환은 이동, 회전으로 제한되나 만약 여러가지 크기의 비품항목이 사용될수 있다면 비례변환을 할수 있다.

8장. 도형사용자대면부 및 대화식입력방법

대부분의 체계들에서 사람-콤퓨터대면부는 응용에 관계없이 여러가지 도형들을 포함하고 있다. 지금 체계들은 일반적으로 모두 창문, 내려펼침 및 튀여나옴차림표, 그림기호, 화면에서 유표의 위치를 지정하는 마우스, 공간볼과 같은 위치지정장치들로 이루어 진다. 널리 쓰이는 도형사용자대면부로서는 X Windows, Windows, Macintosh, OpenLook, Motif가 있다. 이 대면부들은 문서처리, 표처리, 자료기지 및 파일관리체계, 연시체계, 페지설계체계를 비롯한 여러 응용분야들에서 리용된다. 대화식전용도형처리프로그람들은 공학설계, 건축설계, 자료의 가시화, 작도, 업무그라프, 미술가의 그림그리기프로그람과 같이 개별적인 응용에 맞게 설계된다. 일반도형처리프로그람들의 대면부는 보통 표준체계로 만든다. 실례로서 PHIGS에 의한 X Windows System대면부를 들수 있다. 이 장에서는 도형사용자대면부의 기본요소들과 대화기술을 고찰한다. 그리고 특히 도형처리프로그람들에서 대화를 통하여 어떻게 그림요소를 만들고 처리하며 어떻게 차림표항목을 선택하고 파라메터값을 할당하며 어떻게 본문렬을 선택하고 위치를 정하는가를 고찰한다. 여러가지 입력장치들이 있으므로 일반도형처리프로그람들은 이런 장치들과 결합되여 폭 넓은 대화능력을 가지고 있다.

1절. 사용자대화

매개 응용에서 사용자모형은 대화설계의 기초로 된다. 사용자모형은 체계가 무엇을 수행하기 위하여 설계되며 어떤 도형처리조작들을 사용할수 있는가 하는것을 서술한다. 이것은 어떤 형의 물체가 현시될수 있으며 물체가 어떻게 처리될수 있는가를 말한다. 실례로 도형처리체계가 건축설계를 위한 도구로 리용된다면 모형은 벽, 문, 창문 기타 건물요소들의 위치를 지정하여 건물의 보임상을 만들고 현시하는데서 그 설계프로그람을 어떻게 리용할수 있는가를 서술한다. 류사하게 사무실설계체계에서는 물체들이 비품항목(책상, 의자 등)들의 모임으로 정의될수 있고 사용가능한 조작들에는 사무실설계에서 서로 다른 비품들의 위치지정조작과 제거조작이 포함된다. 그리고 회로설계프로그람에서는 물체에 대한 전기 또는 론리요소들이 리용될수 있고 전체 회로설계에서 요소들을 추가하고 지우는데 사용할수 있는 위치지정조작들이 리용될수 있다.

사용자대화의 모든 정보는 응용분야의 언어로 제출된다. 건축설계프로그람이라면 이것은 개별적인 자료구조나 건축가가 잘 모르는 다른 개념들을 참조함이 없이 모든 대화들이 건축용어로만 서술된다는것을 의미한다. 다음부분에서는 사용자대화의 조직과 관련된 일반적인 몇가지 고찰을 진행한다.

창문 및 그림기호

그림 8-1은 창문 및 그림기호가 있는 도형대면부의 일반적인 실례이다. 시각적인 표현은 응용에서 처리될 대상과 그 응용대상에서 수행해야 할 동작에 다같이 리용된다.

ㄱ)

ㄴ)

ㄷ)

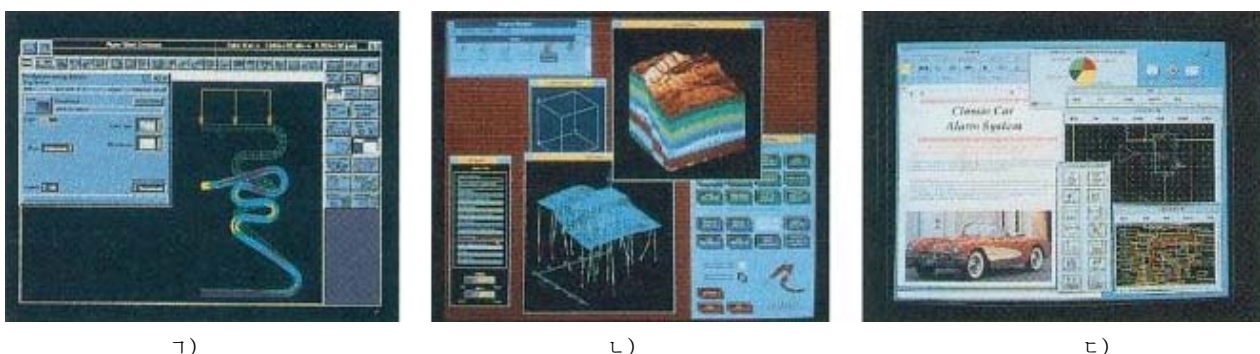



그림 8-1. 창문체계와 그림기호를 리용하는 화면설계의 실례

창문체계는 사용자에게 창문관리대면부와 창문의 현시 및 처리를 조종하는 함수들을 준다. 창문체계의 일반적인 기능은 창문의 열기와 닫기, 창문의 위치를 다시정하기, 창문의 크기를 변경하기, 내부보존 및 외부보존자르기가 있는 현시루틴 기타 다른 도형처리기능들이다. 일반적으로 창문은 미끄럼띠, 누름단추 그리고 여러가지 창문선택항목을 선택하는 차림표그림기호들로 현시된다. X Windows와 NeWS와 같은 일부 일반체계들은 다중창문관리자를 가지고 있으므로 창문별로 자기의 창문관리자에 의하여 서로 다른 창문형태를 만들수 있다. 이때 창문관리자들은 개별적인 응용목적에 맞게 설계된다. 다른 경우에는 창문체계가 하나의 구체적인 응용과 창문형태에 대하여 설계된다.

비품항목과 회로요소와 같은 물체를 표현하는 그림기호를 흔히 **응용그림기호**라고 한다. 회전, 확대, 비례변환, 자르기, 붙이기와 같은 동작을 표현하는 그림기호를 **조종그림기호** 또는 **지령그림기호**라고 한다.

여러 준위의 편의제공

일반적으로 대화식도형대면부들은 동작을 선택하는 여러가지 방법들을 가지고 있다. 실례로 선택항목은 그림기호를 지적하고 마우스단추를 찰칵하거나 또는 내리펼침 또는 튀여나옴차림표를 호출하거나 또는 건지령을 쳐 넣는 방법으로 선택할수 있다. 이것은 각이한 수준의 사용자들에게 편리하다.

경험이 적은 사용자에게는 좀 쉽게 리해되는 조작들과 구체화된 짧은 지시가 있는 대면부가 포괄적인 수많은 조작들의 모임을 가진 대면부보다 더 효과적이다. 차림표와 선택항목들의 모임을 간단화하면 배우고 기억하기 쉬우며 사용자가 대면부의 세부가 아니라 응용에 주의를 집중할수 있게 한다. 응용프로그람에 대한 경험이 있는 사용자에게는 흔히 간단한 지시-찰칵조작이 제일 좋다. 따라서 대면부들은 일반적으로 프로그람의 복잡한 내용을 가리우는 방법들을 제공하여 초학자들이 너무 많은 세부때문에 어리둥절해 짐이 없이 체계를 리용할수 있게 한다.

한편 경험 있는 사용자는 일반적으로 속도를 요구한다. 이것은 약간한 입력재촉과 건반 또는 다중마우스단추의 찰칵에 의하여 더 많이 입력한다는것을 의미한다. 경험 있는 사용자들은 일반적으로 리용되는 동작들에 대한 지름길을 기억하고 있기때문에 동작들을 기능건 또는 건들의 동시결합에 의하여 선택한다.

이렇게 방조편의를 여러 준위로 설계하면 초학자들은 그들대로 구체화된 대화를 진행할수 있고 또 경험 있는 사용자들은 입력재촉과 통보문이 없이 작업할수 있게 된다. 방조편의에는 또한 사용자에게 체계의 능력과 사용에 대한 소개를 주는 하나 또는 그이상의 교육프로그람이 들어 갈수 있다.

일관성

대면부설계에서 중요하게 고려해야 할 문제는 일관성이다. 실례로 매개 그림기호형태는 문맥에 따라 서로 다른 동작이나 물체를 표현하게 하지 말고 항상 한가지 의미를 가지게 해야 한다. 일관성의 몇가지 다른 실례는 사용자가 매개 선택항목을 찾아 다니지 않도록 항상 차림표들을 동일한 상대적위치에 놓는것, 동일한 동작에 대하여 항상 개별적인 건결합을 리용하는것, 항상 색을 부호화하되 동일한 색이 각이한 조건에서 서로 다른 의미를 가지지 않도록 하는것이다.

일반적으로 복잡하고 일관성이 없는 모형은 사용자가 리해하기 힘들고 효과적으로 작업하기 힘들다. 제공되는 물체와 조작들은 적으면서도 모순이 없는 모임을 이룸으로써 체계를 배우기 쉽게 설계되여야 한다. 그러나 여러가지로 쓰일 정도로 지나치게 간단화되지는 말아야 한다.

기억최소화

대면부에서의 조작들은 또한 리해하고 암기하기 쉽도록 조직되여야 한다. 모호하고 복잡하며 모

순이 있고 너무 생략된 지령형식들은 프로그람의 사용에서 혼란을 가져 오거나 효과성을 떨구게 한다. 실례로 모든 지우기조작에 하나의 건이나 단추를 리용하는것이 지우기조작의 종류별로 몇개의 건들을 각각 쓰는것보다 기억하기가 더 쉽다.

그림기호와 창문체계는 암기도 적게 할수 있게 한다. 각이한 종류의 정보들이 서로 다른 창문에 갈라 지므로 서로 다른 정보표시들이 겹칠 때 암기에 의거하지 않아도 된다. 여러가지 정보들을 화면의 서로 다른 창문들에서 간단히 보유할수 있으며 창문구역들사이를 앞뒤로 절환할수 있다. 그림기호들을 여러가지 물체와 동작이 쉽게 인식될수 있는 형태들을 가지게 하면 기억을 줄이는데도 좋다. 매개 동작을 선택하려면 간단히 그 동작을 상징하는 그림기호를 선택하면 된다.

되돌아가기 및 오유처리

대면부의 다른 일반적인 특징은 조작을 진행할 때 되돌아가기동작 또는 중지동작이다. 조작은 흔히 실행이 완성되기전에 취소될수 있으며 이때 체계는 조작이 시작되기전의 상태로 돌아 가야 한다. 임의의 시각에 뒤로 되돌아 갈수 있는 능력이 있으면 틀린 결과를 지울수 있다는것을 알고 체계의 능력을 대담하게 탐구할수 있다.

되돌아가기는 여러가지 형식으로 줄수 있다. 표준적인 취소건 또는 지령은 하나의 조작을 취소하는데 리용된다. 때때로 체계는 여러개 조작들을 되돌아 가게 할수 있으며 체계를 어떤 지적된 곳에 재설정할수 있게 한다. 확장된 되돌아가기능력을 가진 체계에서는 작업중의 모든 입력들을 보관하여 임의의 부분에 되돌아 가게 하거나 다시 할수 있도록 하고 있다.

어떤 조작들은 취소될수 없다. 실례로 탁상휴지통안의 쓰레기는 일단 지우면 지워 진 파일들을 회복할수 없다. 이 경우에 대면부는 집행하기전에 지우기조작에 대한 확답을 요구한다.

오유의 원인결정을 돕도록 하기 위하여 적절한 진단과 오유통보문들이 설계된다. 추가적으로 대면부는 오유로 귀착될수 있는 일정한 동작들을 예측하게 함으로써 오유의 가능성을 최소화하도록 한다. 이런 실례들로는 선택된 물체가 없을 때 물체의 위치를 이동시키거나 물체를 지우지 못하게 하는 것, 선택된 물체가 선이 아닐 때 선의 속성을 선택하지 못하게 하는것, 오려둠판에 아무것도 없을 때 붙이기조작을 선택하지 못하게 하는것들이다.

반결합

대면부는 련속적인 호상작용대화를 진행하여 매 걸음에서 진행중의 동작을 알게 설계된다. 이것은 응답시간이 뜰 때 특히 중요하다. 반결합이 없이는 체계가 무엇을 하고 있는지 그리고 입력이 다시 주어 져야 하는지 궁금해 할수 있다.

매 입력을 받을 때마다 체계는 표준적으로 어떤 형의 응답을 준다. 물체가 밝기강조되거나 그림기호가 나타나거나 또는 통보문이 현시된다. 이것은 입력을 받았다는것을 통지할뿐아니라 체계가 무엇을 하고 있는가를 알게 한다. 처리가 몇초안에 완성될수 없으면 체계의 동작상태를 통지하기 위하여 여러가지 반결합통보문을 현시할수도 있다. 일부 경우에는 이것이 체계가 입력의 요청에 따라 여전히 작업하고 있다는것을 지적하는 깜박거리는 통보문일수도 있다. 체계는 또한 부분적인 결과가 완성될 때 그것을 현시하고 최종적인 현시는 조금 있다가 만들어 지게 할수 있다. 체계는 또한 한 지령이 처리되고 있는 동안 다른 지령 또는 자료를 입력하게 할수 있다.

반결합통보문들은 표준적으로 사고가 중단될 정도로 지나치게 길지 않는 짧은 시간동안에 쭉 훑어 볼수 있게 충분히 명백하게 주어 진다. 기능건에 대한 반결합은 《삑》소리를 내게 하거나 눌리워진 건에 불이 오게 할수 있다. 음성반결합은 화면구역을 리용하지 않는 우점을 가지며 작업구역에서 통보문을 받기 위하여 주의할 필요가 없다. 통보문이 화면에 현시될 때에는 통보문이 있는데를 항상

알수 있도록 고정된 통보문구역이 리용된다. 일부 경우들에는 유표가까이의 작업구역에 반결합통보문을 놓는것이 유리하다. 반결합은 현시된 다른 물체들과 구별하기 위하여 다른 색으로 현시할수도 있다.

체계의 응답속도를 높이기 위하여 반결합기술은 사용하는 장치의 동작특성상 우점을 살리도록 선택될수 있다. 일반적인 라스터반결합기술은 특히 차림표를 선택할 때 화소세기를 반전시키는것이다. 다른 반결합방법들로는 밝기강조, 깜빡거림, 색변화가 있다.

반결합에 특수한 기호들도 쓴다. 실례로 곱하기기호, 찌프린 얼굴, 거절기호들은 흔히 오유를 지적하는데 리용되며 작업중 깜빡임부호는 처리가 진행중에 있다는것을 지적하는데 리용된다. 이런 형태의 반결합은 보다 경험이 있는 사용자들에게는 대단히 효과적일수 있지만 초학자들에게는 체계가 무엇을 하고 있는지 또한 사용자가 다음에 무엇을 입력하여야 하는지를 명백히 지적해 주는 보다 구체화된 반결합이 요구된다.

일부 입력형태에서는 반응반결합이 좋다. 타자한 문자는 오유를 즉시에 검출 및 수정하도록 그것을 입력할 때마다 화면에 현시한다. 단추 및 다이얄입력도 같은 방법으로 반응할수 있다. 다이얄에 의하여 또는 현시되는 크기로 선택되는 스칼라값은 입력값의 정밀성을 검사하기 위하여 보통 화면에 값을 반영시킨다. 자리표점의 선택은 선택된 위치에 나타나는 유표 또는 다른 기호로 반응하게 할수 있다. 선택된 위치를 정확히 보기 위하여 자리표값을 화면에 현시할수 있다.

2절. 도형자료의 입력

도형처리프로그람들에서는 여러가지 종류의 입력자료를 리용한다. 그림지적에는 자리표위치값, 문자렬파라메터들에 대한 값, 변환파라메터들에 대한 수값, 차림표선택항목들을 지적하는 값, 그림부분들의 식별값들이 필요하다. 2장에서 설명한 임의의 입력장치들이 여러가지 형태의 도형자료를 입력하는데 리용될수 있는데 일부 장치들은 다른것들과 달리 일정한 자료형에 더 적합하다. 입력함수들은 도형처리프로그람을 개별적인 하드웨어장치와 독립시키기 위하여 매개 함수를 처리하는 자료종류에 따라 구조화한다. 이 방법은 입력되는 자료의 종류에 따르는 입력장치들의 론리적인 분류를 준다.

입력장치들의 론리적인 분류

여러가지 종류의 입력자료들은 PHIGS와 GKS에서 사용하는 다음의 6가지 론리적인 장치분류로 요약된다.

위치지정장치(LOCATOR) - 자리표위치 (x, y)를 지적하는 장치
획긋기장치(STROKE) - 자리표위치들의 련속을 지적하는 장치
문자렬장치(STRING) - 본문입력을 지적하는 장치
수값장치(VALUATOR) - 수값을 지적하는 장치
선택장치(CHOICE) - 차림표선택항목을 선택하는 장치
잡기장치(PICK) - 그림요소를 선택하는 장치

일부 프로그람들에서는 위치지정장치와 획긋기장치의 동작에 하나의 론리장치를 리용한다. 이때 하나의 자리표위치가 입력되는가 또는 위치들의 흐름이 입력되는가를 지적하기 위한 스위치와 같은 일부 다른 기구가 리용될수 있다.

론리적인 분류에서 6가지 매 입력장치는 어느 하드웨어장치로써도 실현할수 있지만 어떤 하드웨

어장치들은 다른것들보다 일정한 종류의 자료에 더 편리하다. 실례로 화면의 위치를 지적할수 있는 장치는 자리표자료를 입력하는데서 건반보다 더 편리하다. 다음의 부분들에서는 여러가지 물리적인 장치들이 매 형태의 론리적인 입력을 주는데 어떻게 리용되는가를 설명한다.

위치지정장치

자리표점을 대화적으로 선택하는 표준적인 방법은 화면유표의 위치지정에 의한 방법이다. 이것은 마우스, 조종간, 추적볼, 공간볼, 손가락굴리개, 다이얄, 수자화기의 펜이나 손유표 또는 일부 다른 유표위치지정장치에 의하여 진행할수 있다. 화면유표가 요구되는 위치에 있을 때 단추를 동작시켜 화면우의 점의 자리표를 기억시킨다.

건반은 위치지정장치입력에 여러가지 방식으로 리용될수 있다. 일반목적건반은 보통 화면유표를 우, 아래, 왼쪽, 오른쪽으로 움직이는 4개의 유표조종건을 가지고 있다. 또한 4개의 추가적인 건에 의하여 유표을 대각선방향으로 움직일수 있다. 빠른 유표이동은 선택된 유표건을 계속 누르는 방법으로 진행한다. 또 다르게 상대적인 유표이동을 위하여 조종간, 추적볼 또는 손가락굴리개가 건반에 설치될수 있다. 마지막수단으로 자리표값을 실제로 쳐 넣을수도 있는데 이것은 정확한 자리표값을 알아야 하는 보다 느린 처리이다.

빛펜 역시 자리표위치입력에 리용되지만 실현에서 특별히 고려해야 할 일부 문제들이 있다. 빛펜은 화면형광체로부터 방출되는 빛을 검출하여 동작하기때문에 선택되는 자리표위치에 어떤 세기의 배경준위가 있어야 한다. 라스터체계에서는 화면에 색배경을 칠할수 있다. 새까만 구역이 없는한 빛펜은 임의의 화면위치를 선택하는데 리용될수 있다. 현시장치에서 모든 새까만 구역들을 없애는것이 불가능할 때(실례로 벡토르체계에서와 같이)에는 펜을 검출하는 작은 빛무늬를 만드는 방법으로 빛펜을 위치지정장치로 리용할수 있다. 빛무늬는 빛펜을 찾을 때까지 화면주위에서 움직인다.

획긋기장치

이 부류의 론리장치는 자리표위치들의 순서렬을 입력하는데 리용한다. 획긋기장치의 입력은 위치지정장치를 여러번 호출하는것과 같다. 입력점들의 모임은 흔히 선분을 현시하는데 리용한다.

위치지정장치입력을 발생시키는데 리용되는 많은 물리적인 장치들은 획긋기장치로 리용할수 있다. 마우스, 추적볼, 조종간, 도형입력판의 손유표의 련속적인 이동은 입력자리표값들의 련속으로 번역된다. 도형입력판은 보다 일반적인 획긋기장치들중의 하나이다. 단추를 동작시켜 도형입력판을 련속방식으로 놓을수 있다. 유표가 도형입력판면을 따라서 움직일 때 자리표값들의 흐름이 발생된다. 이 처리는 미술가가 화면에서 장면을 그릴수 있게 하는 그림그리기체계 그리고 도면을 추적하고 수자변환하여 기억시키기 위한 공학체계들에서 리용된다.

문자렬장치

문자렬입력에 리용된 최초의 물리적인 장치는 건반이다. 입력문자렬은 일반적으로 그림이나 그라프의 표식으로 리용된다.

다른 물리적인 장치들은 《본문을 쓰는 방식》으로 문자형태를 발생시키는데 리용될수 있다. 이런 입력에 의한 개별적인 문자들은 획긋기 또는 위치지정장치에 의하여 화면에 그려 진다. 그러면 형태인식프로그람이 미리 정의된 형태들이 기억된 사전을 리용하여 문자들을 통역한다.

수값장치

이 부류의 론리장치는 도형처리체계들에서 수값을 입력하는데 리용한다. 수값은 회전각이나 크기

와 같은 여러가지 도형처리파라메터의 설정, 매개 응용과 관련되는 물리적인 파라메터(온도, 전압준위, 압력 등)설정에 리용한다.

수값입력에 리용되는 대표적인 물리적인 장치는 조종다이얄들의 모임이다. 임의의 미리 정의된 범위안의 류점수를 다이얄을 회전시켜 입력한다. 한쪽 방향으로 눈금판을 돌리면 입력수값이 증가하고 반대방향으로 돌리면 수값이 감소한다. 가변저항은 다이얄회전을 대응하는 전압으로 변환한다. 이 전압은 다음에 실례로 -10.5로부터 25.5까지와 같이 이미 정의된 크기범위안의 실수로 번역된다. 때때로 다이얄대신에 미끄럼가변저항을 리용하여 직선적인 이동을 수값으로 변환하기도 한다. 수자건들의 모임을 가지고 있는 모든 건반은 수값장치로 리용될수 있다. 사용자는 수를 류점수형식으로 직접 간단히 쳐넣을수 있지만 이것은 다이얄이나 미끄럼가변저항를 리용하는것보다 느린 방법이다.

조종간, 추적볼, 도형입력판, 기타 대화식장치들은 장치의 압력 또는 이동을 크기값범위로 변환함으로써 수값입력에 적응시킬수 있다. 한쪽 방향의 이동 례를 들면 왼쪽에서 오른쪽으로의 이동에서는 입력되는 수값이 증가되고. 반대반향으로의 이동에서는 입력되는 수값이 감소된다.

수값을 입력하는 다른 수법은 현시장치에 미끄럼띠, 단추, 회전눈금자, 차림표를 현시하는것이다. 그림 8-2는 눈금자를 표현하는 일부 수법들을 보여 주었다. 마우스, 조종간, 공간볼 또는 기타 다른 장치로부터의 위치지정장치입력이 현시장치에서의 자리표위치를 선택하는데 리용된다.이 화면자리표위치가 그다음 입력수값으로 변환된다. 눈금자에서 선택되는 위치에는 어떤 기호에 의하여 사용자에 대한 반결합처럼 표식을 매길수 있다. 수값의 정확성을 확인하기 위하여 화면의 어떤 곳에 그 수값을 현시할수 있다.

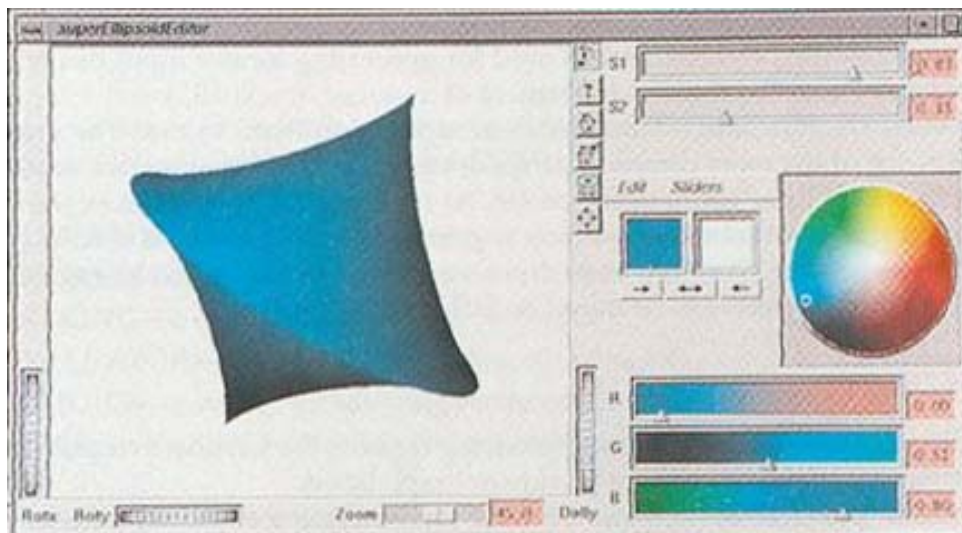



그림 8-2. 파라메터값들을 대화식으로 선택하기 위하여 현시장치에 현시한 눈금자(이 화면에서 미끄럼눈금자들은 초타원파라메터 s1과 s2의 크기와 개별적인 R, G, B색값을 선택하는데 쓴다. 그리고 작은 색원안에서는 위치를 지정하여 RGB합성색을 선택하며 단추로는 색값의 작은 변화를 지적한다.)

선택장치

도형처리프로그람들은 프로그람작성의 선택항목, 파라메터값, 그림을 만드는데 리용될 물체형태들을 선택하는데 차림표를 리용한다(그림 8-1). 선택장치는 선택항목들의 목록(차림표)으로부터 선택한것을 입력하는 장치이다. 일반적으로 리용되는 선택장치에는 단추들의 모임, 마우스, 추적볼, 건반의 유표건과 같은 유표위치지정장치 그리고 접촉판이 있다.

독립적인 장치로 설계된 기능건반 또는 《단추통》이 흔히 차림표선택을 입력하는데 리용된다. 보

통 매개 단추는 프로그람화할수 있다. 그러므로 그 기능들을 변경시키면 다른 응용들에 적응시킬수 있다. 한가지 목적의 단추는 미리 정의된 고정된 기능을 가진다. 흔히 건반에는 프로그람화할수 있는 기능건들과 고정된 기능의 단추들이 다른 표준적인 건들과 함께 들어 있다.

목록에 있는 차림표선택항목들은 유표조종장치들을 리용하여 화면에서 선택할수 있다. 자리표위치 (x, y)가 선택되면 그것은 목록의 매개 차림표항목들의 자리표범위와 비교된다. 수평 및 수직경계자리표값들이 x_{min}, x_{max}, y_{min}, , y_{max}인 차림표항목은 입력자리표 (x, y)가 부등식

$$x_{min} \le x \le x_{max}, \qquad y_{min} \le y \le y_{max} \tag{8-1}$$

을 만족시키면 선택된다.

한번에 여러개의 선택항목을 현시하는 큰 차림표에서는 일반적으로 다침판이 리용된다. 마우스와 같은 유표조종장치에서와 마찬가지로 선택되는 화면의 위치는 매개 차림표선택항목이 차지하는 구역과 비교된다.

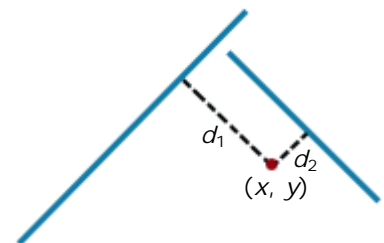
또 다른 선택입력방법에는 건반과 음성입력이 있다. 표준적인 건반은 지령이나 또는 차림표선택항목을 타자하는데 리용할수 있다. 이 방법의 선택입력에서는 생략된 형식을 쓰면 편리하다. 이를 위하여 차림표목록에 번호를 매겨 놓거나 또는 짧은 식별이름을 준다. 이러한 부호화는 음성입력체계에서도 리용할수 있다. 음성입력은 특히 선택항목의 개수가 작을 때(20 또는 이하)에 쓸모 있다.

잡기장치

도형물체를 선택하는것이 이 부류의 론리장치의 기능이다. 잡기장치는 어떤 방식으로 변환하거나 편집할 장면의 부분을 선택하는데 리용된다.

물체선택에 리용되는 일반적인 장치는 차림표선택에 대한것과 같다. 즉 유표위치지정장치이다. 마우스 또는 조종간으로 현시된 구조물내의 기초요소들에 유표의 위치를 지정하고 선택단추를 누를수 있다. 그러면 유표의 위치를 기록하고 선택할 개별적인 물체를 찾아 내기 위하여 여러가지 준위의 탐색을 진행한다. 먼저 유표의 위치를 장면안의 여러 구조물들의 자리표범위와 비교한다. 한 구조물의 경계직4각형이 유표의 자리표를 포함하면 잡기구조물로 판정한다. 그러나 둘 또는 그이상의 구조물들의 구역들이 유표자리표를 포함하면 한걸음 더 나아가 검사를 해야 한다. 이번에는 매개 구조물에 있는 개별적인 선들의 자리표범위를 검사할수 있다. 실례로 유표자리표가 하나의 선의 자리표범위내에만 있다고 결정되면 잡기물체로 식별된다. 그렇지 않으면 유표위치에 제일 가까운 선을 결정하는 추가적인 검사를 하게 된다.

그림 8-3. 선토막으로부터 잡기위치까지의 거리

유표위치에 제일 가까운 선을 찾는 한가지 방법은 유표자리표 (x, y)로부터 유표위치를 포함하는 경계직4각형의 매개 선토막까지의 거리의 두제곱을 계산하는것이다(그림 8-3). 끝점 (x_1, y_1)과 (x_2, y_2)를 가지는 선에 대하여 (x, y)로부터 선까지의 거리의 두제곱은

$$d^2 = \frac{[\Delta x(y - y_1) - \Delta y(x - x_1)]^2}{\Delta x^2 + \Delta y^2} \tag{8-2}$$

와 같이 계산된다. 여기서 $\Delta x = x_2 - x_1$ 그리고 $\Delta y = y_2 - y_1$이다. 이 거리계산의 속도를 높이기 위하여 여러가지 근사식들을 리용할수도 있고 다른 식별방안들을 리용할수 있다.

유표의 위치에서 제일 가까운 선을 찾는 다른 방법은 잡기창문에 크기를 주는것이다. 유표자리표는 이 창문의 중심에 있으며 그림 8-4에 보여 준바와 같이 후보선들은 창문에 의하여 잘라 지게 된

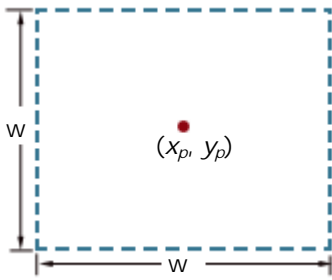



그림 8-4. 잡기물체들의 겹침을 해결하는데 리용되는 잡기자리표 (x_p, y_p)에 중심이 있는 잡기창문

다. 잡기창문을 충분히 작게 하면 하나의 선만이 창문을 지나가게 할수 있다. 잡기창문의 크기를 선택하는 방법은 여러가지 입력함수들과 관련되는 파라메터들을 고찰하는 8장 4절에서 서술한다.

잡기거리 또는 창문자르기사귐점계산을 피하는 방법은 선택될수 있는 구조물들을 밝기강조하여 사용자가 잡기에서의 모호함을 해결하게 하는것이다. 이렇게 하는 한가지 방법은 유표위치와 겹치는 구조물들을 하나하나씩 밝기강조하는것이다. 이때 사용자는 요구되는 구조물이 밝기강조되였을 때에 신호한다.

다른 방법은 유표의 위치지정과는 달리 단추입력을 리용하여 련속적으로 구조물들을 밝기강조하는것이다. 두번째 단추는 요구되는 구조물이 밝기강조되였을 때 처리를 중지시키는데 리용한다. 이 방법으로 대단히 많은 구조물들이 탐색된다면 처리속도를 높일수 있다. 추가적인 단추를 리용하여 구조물식별을 돕는다. 첫번째 단추로는 구조물들의 빠른 련속적인 밝기강조를 시작할수 있다. 두번째 단추는 역시 처리를 중지시키는데 리용할수 있으며 세번째 단추는 조작자가 중지단추를 누르기전에 요구되는 구조물이 지나갔으면 보다 천천히 역행시키는데 리용할수 있다.

마지막으로 건반을 리용하여 구조물의 이름을 타자할수 있다. 이것은 간단은 하지만 대화성이 부족한 잡기선택방법이다. 서술적인 이름은 사용자의 잡기처리를 돕는데 리용될수 있지만 이 방법은 여러가지 결함들을 가지고 있다. 그것은 일반적으로 화면에서의 대화식잡기보다 느리며 사용자가 여러가지 구조물들의 이름을 기억하기 위한 대책이 필요하기때문이다. 게다가 건반으로부터 구조물의 일부분을 잡는것은 화면에서 일부분을 잡는것보다 더 힘들수 있다.

3절. 입력함수

도형입력함수는 사용자가 다음의 선택항목들을 지적하도록 설정될수 있다.

- 어떤 물리적인 장치가 론리적인 분류안에서의 어떤 입력을 주려고 하는가(실례로 도형입력판을 획긋기장치로 리용한다.).
- 도형처리프로그람과 장치가 어떻게 호상작용하는가(입력방식). 프로그람 또는 장치중 어느 하나가 자료입력을 시작할수도 있고 둘이 다 동시에 동작할수도 있다.
- 매개 입력형을 지적된 자료변수에 넘겨 주기 위하여 자료가 언제 입력되며 어느 장치가 그 시각에 리용되는가.

입력방식

입력을 주는 함수들은 프로그람과 입력장치가 어떻게 대화하는가를 지적하는 여러가지 입력방식들에서 동작하도록 구조화될수 있다. 입력은 프로그람에 의하여 시작될수도 있고 프로그람과 입력장치가 다같이 동시에 동작할수도 있으며 자료입력이 장치에 의하여 시작될수도 있다. 이 세가지 입력방식들을 요청방식, 표본방식, 사건방식이라고 한다.

요청방식에서는 응용프로그람이 자료입력을 시작한다. 입력값이 요청되면 요구하는 값을 받을 때

까지 처리는 중지된다. 이 입력방식은 일반적인 프로그람작성언어들에서 공통적인 입력조작에 대응된다. 프로그람과 입력장치가 번갈아 동작한다. 장치는 입구요청이 있을 때까지 대기상태에 들어 가며 프로그람은 자료를 넘겨 줄 때까지 대기한다.

표본방식에서는 응용프로그람과 입력장치가 독립적으로 동작한다. 입력장치는 프로그람이 다른 자료를 처리하고 있는 그 시각에도 동작할수 있다. 입력장치로부터의 새로운 입력값은 앞서 입력된 자료값을 교체하여 기억된다. 프로그람은 새로운 자료가 요구되면 입력장치로부터 현재의 값을 표본으로 취한다.

사건방식에서는 입력장치가 응용프로그람에 대하여 자료넣기를 시작한다. 프로그람과 입력장치는 역시 동시에 동작하지만 이때에는 입력장치가 자료를 입력대기렬에 넘겨 준다. 모든 입력자료가 보관된다. 프로그람이 새로운 자료를 요구할 때 그것은 자료대기렬에 넘어 간다.

표본 및 사건방식에서는 임의의 개수의 장치들이 동시에 동작할수 있다. 일부는 표본방식에서 동작하고 다른것들은 사건방식에서 동작할수 있다. 그러나 요청방식에서는 한번에 오직 하나의 장치만이 입력을 줄수 있다.

지적된 워크스테이션으로 동작하는 매개 물리적인 장치에 대한 론리적인 분류에서의 입력방식은

```
set … Mode (ws, deviceCode, inputMode, echoFlag)
```

형식의 6가지 부류의 입력함수중 하나로 선언된다. 여기서 deviceCode는 정의옹근수이며 inputMode에는 값 request(요청), sample(표본), event(사건)중의 하나가 할당되며 파라메터 echoFlag에는 값 echo(반영) 또는 noecho(반영없음)가 할당된다. 입력자료가 현시장치에 어떻게 반영되는가는 이 절의 뒤부분에서 서술되는 다른 입력함수들에 설정되는 파라메터들에 의해 결정된다.

장치코드 deviceCode의 할당은 설치에 관계된다. 장치코드의 한가지 가능한 할당을 표 8-1에 보여 주었다. 이 표의 값들을 리용하여 다음의 선언을 할수 있다.

```
setLocatorMode(1, 2, sample, noecho)
setTextMode(2, 1, request, echo)
setPickMode(4, 3, event, echo)
```

따라서 도형입력판은 워크스테이션1로서 입력자료의 반결합반영이 없이 표본방식의 위치지정장치로 선언되고 건반은 워크스테이션2로서 입력반영이 있는 요청방식의 본문장치로 그리고 마우스는 워크스테이션4로서 입력반영이 있는 사건방식의 잡기장치로 선언되였다.

표 8-1. 입력장치코드의 할당

장치코드	물리적인 장치형
1	건반
2	도형입력판
3	마우스
4	조종간
5	추적볼
6	단추통

요청방식

이 방식에서 사용하는 입력지령들은 고수준프로그람작성언어들에서의 표준적인 입력함수들에 대응된다. 요청방식으로 입력을 요구할 때에는 입력을 받을 때까지 다른 처리를 중지한다. 장치에 요청방식이 할당된후 앞에서 설명한바와 같이 해당 장치에 대한 입력요청은 다음과 같이 표현되는 6가지의 론리부류의 함수중 하나를 리용하여 진행할수 있다.

```
request … (ws, deviceCode, status, … )
```

이 함수에 넘겨 주는 값은 워크스테이션코드와 장치코드이다. 되돌림값은 파라메터 status와 요청되는 론리적부류의 대응하는 자료파라메터들에 할당된다.

입력자료의 유효성에 따라 파라메터 status에는 값 ok 또는 none가 되돌려 진다. 값 none는 입력장치가 무효한 자료를 만들고 있다는것을 지적한다. 위치지정장치의 입력에 대하여 말하면 이것은 자리표가 범위밖에 있다는것을 의미한다. 잡기입력에 대하여서는 장치는 동작하였지만 구조물을 가리키지 않을수 있다. 또는 입력장치에서 break단추가 눌리워 졌을수도 있다. 되돌림값 none는 자료끝신호로 리용될수도 있다.

요청방식에서 위치지정장치 및 획긋기장치의 입력

이 두 론리입력부류에 대한 요청함수는

```
requestLocator(ws, deviceCode, status, viewIndex, pt)
requestStroke(ws, deviceCode, nMax, status, viewIndex, n, pts)
```

이다. 위치지정장치입력에서 pt는 선택된 세계자리표위치이다. 획긋기장치입력에서 pts는 n개 자리표위치들의 목록이며 파라메터 nMax는 입력목록에 들어 갈수 있는 점들의 최대개수이다. 파라메터 viewIndex에는 2차원보기첨수번호가 할당된다.

세계자리표위치의 결정은 두 걸음으로 진행되는 과정이다.

(1) 물리적인 장치가 장치자리표의 점을 선택하면(보통 현시장치화면으로부터) 워크스테이션역변환을 진행하여 정규장치자리표의 대응하는 점을 얻는다.
(2) 다음에 보기자리표 그다음 세계자리표에 도달하기 위하여 창문-보임창역넘기기가 진행된다.

한 장치에서 두개 또는 그이상의 보임상이 겹칠수 있기때문에 정확한 보기변환은 보기변환입력우선권번호에 따라 식별한다. 기정값은 보기첨수번호와 같으며 번호가 낮을수록 우선권은 더 높다. 보기첨수 0이 제일 높은 우선권을 가진다. 보기우선권은

```
setViewTransformationInputPriority(ws, viewIndex,
                              refViewIndex, priority)
```

에 의하여 다른보기변환(참조보기변환)에 대하여 상대적으로 변화시킬수 있다. 여기서 viewIndex는 우선권을 변화시킬 보기변환이며 refViewIndex는 참조보기변환이다. 파라메터 priority에는 값 lower(낮게) 또는 higher(높게)가 할당된다. 실례로 그림 8-5에 보여 준바와 같이 워크스테이션1에서 첫 4개의 보기변환들의 우선권을 함수렬

```
setViewTransformationInputPriority(1, 3, 1, higher)
setViewTransformationInputPriority(1, 0, 2, lower)
```

에 의하여 변경시킬수 있다.

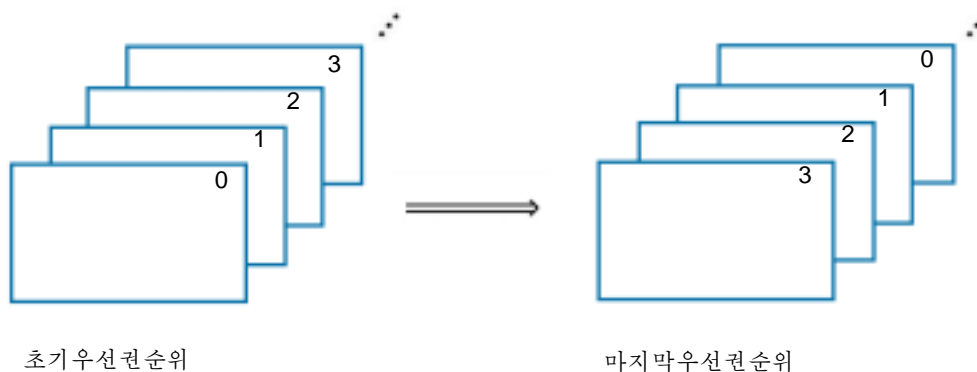



그림 8-5. 보기우선권들의 재정돈

요청방식에서 문자렬입력

여기서의 요청입력함수는

```
requestString(ws, devCode, status, nChars, str)
```

이다. 이 함수에서 파라메터 str에는 입력문자렬이 할당된다. 문자렬에서 문자의 개수는 파라메터 nChars에 주어 진다.

요청방식에서 수값입력

수값은 요청방식에서

```
requestValuator(ws, devCode, status, value)
```

에 의하여 입력된다. 파라메터 value에는 임의의 실수값을 준다.

요청방식에서 선택입력

다음의 요청함수에 의하여 차림표를 선택한다.

```
requestChoice(ws, devCode, status, itemNum)
```

파라메터 itemNum에는 선택된 차림표항목에 대응하는 정의옹근수값을 준다.

요청방식에서 잡기입력

이 방식에서 구조물의 식별자번호는 함수

```
requestPick(ws, devCode, maxPathDepth, status, pathDepth, pickPath)
```

에 의하여 얻는다. 파라메터 pickPath는 선택되는 기초요소를 식별하는 정보들의 목록이다. 이 목록에는 구조물의 이름, 기초요소에 대한 잡기식별자, 원소의 순서렬번호가 포함된다. 파라메터 pickDepth는 pickPath에 주는 준위들의 개수이며 maxPickDepth는 pickPath에 포함될수 있는 최대경로길이를 준다.

구조물의 부분들은 잡기입력에 대하여 다음의 함수로 표식할수 있다.

```
SetPickIdentifier(pickID)
```

구조물생성시 부분들에 대한 표식의 실례를 다음의 프로그람토막에 주었다.

```
openStructure(id);
   for (k = 0; k < n; k++){
          setPickIdentifier (k);
              .
          .
          .
   }
closeStructure;
```

구조물과 구조물의 부분에 대한 잡기는 워크스테이션려파에 의하여서도 일부 조종한다(7장 1절). 물체는 보이지 않으면 잡을수 없다. 또한 보임성에 관계없이 물체를 잡게 할수 있다. 이것은 잡기려파함수

```
setPickFilter(ws, devCode, pickables, nonpickables)
```

에 의하여 수행한다. 여기서 모임 pickables에는 지적된 잡기장치로 선택할것을 바라는 물체(구조물과 기초요소)들의 이름이 포함된다. 류사하게 모임 nonpickables에는 이 입력장치로 잡지 말아야 할 물체들의 이름이 포함된다.

표본방식

하나 또는 그이상의 물리적인 장치들에 대하여 표본방식이 설정되면 자료입력이 프로그람의 지휘를 기다리지 않고 시작된다. 조종간이 표본방식의 위치지정장치로 지적되면 동작상태의 조종간의 현재위치의 자리표값이 즉시에 기억된다. 동작상태의 조종간의 위치가 변하면 기억된 값은 조종간의 현재위치의 자리표로 계속 교체된다.

이 방식에서 물리적인 장치로부터의 현재값의 표본화는 응용프로그람에서 표본명령을 만날 때에 시작된다. 위치지정장치는 다음과 같은 6가지 함수중 하나에 의하여 표본화된다.

```
Sample … (ws, deviceCode, … )
```

표본방식에서 status파라메터는 일부 론리장치들은 가지지만 일부는 가지지 않는다. 다른 입력파라메터들은 요청방식에서와 같다.

표본입력의 실례로 선택된 물체를 이동 및 회전시킨다고 하자. 다음의 명령문들에서 보여 주는바와 같이 물체에 대한 마지막이동위치는 위치지정장치에 의하여 얻을수 있으며 회전각은 수값장치에 의하여 넣어 줄수 있다.

```
sampleLocator(ws1, dev1, viewIndex, pt)
sampleValuator(ws2, dev2, angle)
```

사건방식

입력장치를 사건방식에 놓는 경우에는 프로그람과 장치가 동시에 동작한다. 장치로부터의 자료입력은 사건대기렬 또는 입력대기렬에 축적된다. 사건방식으로 열려 진 모든 입력장치들은 자료(사건이라고 한다.)를 하나의 사건대기렬에 넣는다. 매개 장치는 자료값이 만들어 지면 그것을 넣는다. 임의의 어떤 시각에 사건대기렬에는 여러가지 자료형들이 입력된 순서로 들어 있을수 있다. 대기렬에 넣어 진 자료는 론리적부류, 워크스테이션번호, 물리적인 장치코드에 의해 식별된다.

응용프로그람에서는 모든 입력에 대한 사건대기렬을 함수

```
awaitEvent(time, ws, deviceClass, deviceCode)
```

에 의하여 검사하도록 지시하게 된다. 파라메터 time은 응용프로그람에 대한 최소의 대기시간을 설정하는데 리용된다. 대기렬이 비여 있으면 time에서 지적된 시간이 지날 때까지 또는 입력이 도착할 때까지 처리는 중지된다. 자료값이 입력되기전에 대기시간이 끝나면 파라메터 deviceClass에는 값 none이 할당된다. time이 값 0으로 되면 프로그람은 대기렬을 검사한다. 그것이 비여 있으면 즉시 다른 처리로 넘어 간다.

만약 awaitEvent함수에 의하여 사건대기렬을 검사하여 대기렬이 비여 있지 않으면 대기렬안의 첫번째 사건이 현재사건기록에 전송된다. 이 입력을 만든 위치지정장치 또는 획긋기장치와 같은 매개 론리적인 장치들은 파라메터 deviceClass에 기억된다. 입력을 만든 매개 워크스테이션과 물리적인 장치를 식별하는 코드는 각각 파라메터 ws와 deviceCode에 기억된다.

현재 사건기록으로부터 자료입력을 꺼내보기 위하여서는 사건방식입력함수를 리용한다. 사건방식에서의 함수들은 요청 및 표본방식의 함수들과 류사하다. 그러나 워크스테이션 및 장치코드파라메터들은 지령에 쓰지 않아도 된다. 왜냐하면 이 파라메터들에 대한 값은 자료기록에 기억되기때문이다. 사용자는 자료를

```
get … ( … )
```

에 의하여 꺼낸다. 실례로 위치지정장치입력을 요구하기 위하여 함수

```
getLocator(viewIndex, pt)
```

를 호출한다.

다음의 프로그람토막에서는 awaitEvent 및 get함수들의 사용실례를 주었다. 워크스테이션1에서 도형입력판(장치코드 2)으로부터 점들의 모임을 입력하고 입력자리표들을 련결하여 선분들의 련속을 현시하게 된다.

```
setStrokeMode (1, 2, event, noecho);

do {
 awaitEvenb (0, ws, deviceClass, deviceCode)
} while (deviceClass != stroke);
```

```
getStroke (nMax, viewlndex, n, pts);
polyline (n, pts);
```

do-while순환은 대기렬에 있을수 있는 다른 장치들에서 오는 자료들을 우회한다. 도형입력판이 사건방식에서 유일하게 열려 진 입력장치이면 이 순환은 필요 없다.

몇개의 장치를 사건방식으로 동시에 리용하면 현시에서 대화식처리를 빨리 할수 있다. 다음의 명령문들은 도형입력판으로부터 입력되는 선들을 단추통에 의하여 지적되는 속성으로 현시한다.

```
SetPolylineIndex (1);
/* set tablet to stroke device, event mode */
setStrokeMode (1, 2, event, noecho);

/* set buttons to choice device, event mode */
setChoiceMode (1, 6, event, noecho);

do {
 awaitEvent (60, ws, deviceClass, deviceCode);
 if (deviceClass == choice) {
  gefcChoice (status, option);
  setPolylineIndex (option) ;
 }
 else
  if (deviceClass == stroke) {
   getStroke (nMax, viewlndex, n, pts);
   polyline (n, pts);
   }
} while (deviceClass != none);
```

일부 추가적인 함수들이 사건방식에 쓰인다. 사건대기렬을 지우는 함수들은 처리가 끝나고 새로운 응용이 시작될 때 편리하다. 이 함수들은 전체 대기렬을 지우거나 또는 지적된 입력장치 및 워크스테이션과 관련되는 자료들만을 지우도록 설정될수 있다.

입력방식들의 동시사용

서로 다른 방식의 입력장치들을 동시에 사용하는 실례를 다음의 프로그람에 주었다. 물체는 마우스로 화면에서 끌고 다닌다. 마지막위치가 선택되면 단추를 눌러 물체의 이동을 끝낸다. 마우스의 위치는 표본방식으로 얻어 지며 단추입력은 사건대기렬에 보내진다.

```
/* drags object in response to mouse input */
/* terminate processing by button press */
setLocatorMode (1, 3, sample, echo);
setChoiceMode (1, 6, event, noecho);
do {
```

```
    sampleLocator (1, 3, viewlndex, pt);

    /* translate object centroid to position pt and draw */

    awaitEvent (0, ws, class, code);
  } while (class != choice);
```

4절. 입력장치파라메터들에 대한 초기값

initialize함수를 리용하여 매 부류의 론리장치들에 여러가지 파라메터들을 설정할수 있다.

```
Initialize ... (ws, deviceCode, ..., pe, coordExt, dataRec)
```

파라메터 pe는 입력재촉 및 반결합형이다. 파라메터 coordExt에는 4개의 자리표값들의 모임이 할당되고 파라메터 dataRec에는 여러가지 조종파라메터들이 기록된다.

위치지정장치입력에 대하여 입력재촉 및 반결합파라메터에 할당할수 있는 일부 값들로는

pe = 1 : 정의된 설치
pe = 2 : 현재위치에 중심이 있는 곱하기모양의 유표
pe = 3 : 초기위치로부터 현재위치까지의 선
pe = 4 : 초기점과 현재위치에 의하여 정의되는 직4각형

이다. 여러가지 다른 선택항목들도 또한 사용할수 있다.

구조물잡기에는 여러가지 다른것들과 함께 다음의 선택항목들도 있다.

pe = 1 : 잡은 기초요소들을 밝기강조한다.
pe = 2 : 잡기값 id를 가지는 모든 기초요소들을 밝기강조한다.
pe = 3 : 전체 구조물을 밝기강조한다.

입력자료에 대한 반영이 요청될 때에는 그것을 파라메터 coordExt의 4개 자리표에 의하여 지적되는 경계직4각형안에 현시한다. 또한 추가적인 선택항목들을 파라메터 dataRec에 설정할수도 있다. 실례로 다음의것들중 임의의것을 다른 선택항목들과 함께 설정할수 있다.

- 잡기창문의 크기
- 최소잡기거리
- 유표현시의 형과 크기
- 잡기조작시 구조물의 밝기강조의 형
- 수값입력에 대한 범위(최소와 최대)
- 수값입력에 대한 분해능(척도)

5절. 대화식그림생성기술

대화식그림생성을 돕기 위하여 도형처리프로그람들에 병합되는 몇가지 기술들이 있다. 위치지정장치 또는 획긋기장치에 의하여 입력된 자리표정보를 선택된 항목에 따라 조정 또는 해석할수 있는 여러가지 입력선택항목들을 제공할수 있다. 실례로 모든 선들을 수평 또는 수직선으로만 제한할수 있다. 입력자리표들은 그리려는 물체의 위치나 경계를 확정할수도 있으며 앞서 현시된 물체를 재정렬하는데 리용할수도 있다.

기본적인 위치지정방법

위치지정장치로 입력되는 자리표값들은 흔히 위치지정방법들에 리용되여 물체 또는 문자렬을 현시하는 위치를 지적한다. 자리표위치들은 지시장치로 보통 화면유표의 위치를 지정하는 방법에 의해 대화적으로 선택한다. 물체 또는 본문렬의 위치지정이 어떻게 수행되는가는 선택된 선택항목에 관계된다. 실례로 본문렬에 대한 화면의 점은 문자렬의 중심위치, 시작위치, 끝위치 또는 4장에서 론의한 문자렬위치지정의 임의의 다른 선택항목으로 취할수 있다. 선인 경우 선분은 두개의 선택된 화면위치사이에서 현시될수 있다.

물체위치지정을 방조하기 위하여 선택된 위치에 대한 수값이 화면에 반영될수 있다. 반영된 자리표값을 안내자로 하여 정밀하게 위치를 지정하도록 선택된 위치를 조정할수 있다.

제한조치

일부 응용들에서는 어떤 형태의 정해 진 방향이나 물체의 배렬를 요구한다. 제한조치는 현시되는 자리표가 지적된 방향 또는 배렬을 만들도록 입력자리표값들을 변경시키는 조치이다. 여러가지 종류의 제한조치기능들이 있지만 제일 일반적인 제한조치는 직선을 수평 또는 수직으로 정렬시키는것이다. 그림 8-6과 그림 8-7에 보여 준 이 형태의 제한조치는 회로망을 설계하는데 편리하다. 이 제한조치에 의해 끝점자리표들이 정확히 지적되였는가에 대해 신경을 쓰지 않아도 수평 또는 수직선을 만들수 있다.

수평 또는 수직제한조치는 임의의 두 입력끝점자리표들이 수평선에 더 가까운가 또는 수직선에 더 가까운가를 결정하는 방법으로 실현한다. 만약 두 끝점의 y값의 차이가 x값의 차이보다 더 작으면 수평선을 현시한다. 그렇지 않으면 수직선을 그린다. 다른 종류의 제한조치들은 자리표들을 입력하여

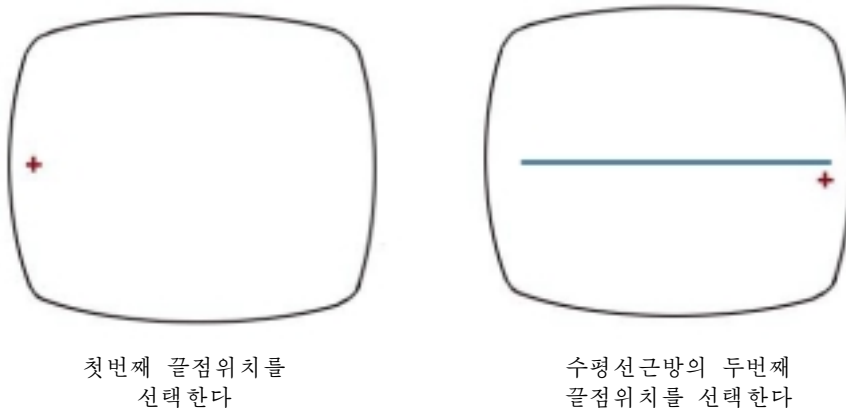



그림 8-6. 수평선제한조치

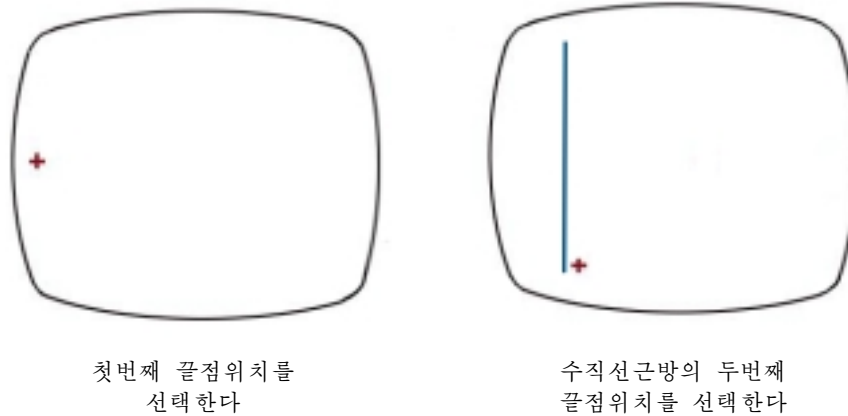



그림 8-7. 수직선제한조치

여러가지로 정렬시키는데 적용한다. 선들이 45° 경사를 가지도록 제한할수 있으며 입력자리표들이 원호와 같은 미리 정의된 경로를 따라 놓이도록 제한할수 있다.

격자

다른 제한은 화면구역의 일부분에 현시되는 직4각형의 격자이다. 격자를 리용하면 모든 입력자리표의 위치는 두 격자선의 제일 가까운 사귐점에 둥그려 진다. 그림 8-8은 격자에 의한 선긋기를 설명한다. 매개 두 유표위치는 제일 가까운 격자사귐점에로 밀려 지며 선은 이 격자점들사이에 그어 진다. 격자는 물체만들기를 쉽게 한다. 왜냐하면 새로운 선은 현시된 선의 한끝의 격자사귐점가까이의 임의의 위치를 선택하여도 앞서 그어 진 선에 쉽게 련결될수 있기때문이다.

격자선들사이의 간격은 흔히 사용자가 설정할수 있는 선택항목이다. 류사하게 격자는 on 및 off로 절환될수 있으며 때때로 서로 다른 화면구역들에 서로 다른 크기의 격자를 리용할수 있다.

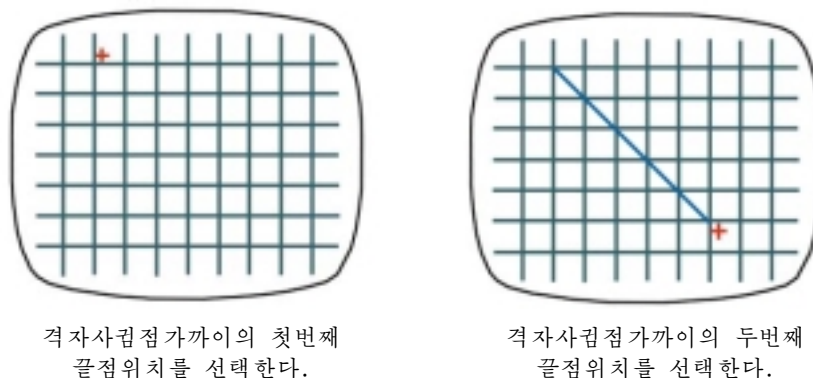



그림 8-8. 격자를 리용한 선긋기

련결마당

그림생성에서는 때때로 선들을 끝점들사이의 위치들에서 련결해야 할 경우도 있다. 화면유표를 련결점에 정확히 위치지정하는것이 힘들수 있기때문에 도형처리프로그람들은 선에 가까운 임의의 입력위치를 그 선의 위치에로 변환하도록 설계한다.

그림 8-9. 선주위의 인력마당 (그늘 진 구역에서 선택되는 임의의 점은 선의 위치에로 끌려 간다.)

입력위치의 이 변환은 선주위에 인력마당구역을 만들어 수행한다. 선의 인력마당안에서 선택되는 임의의 위치는 그 선우의 제일 가까운 위치에로 움직여 진다(끌리운다). 선주위의 인력마당구역을 그림 8-9에 그늘진 경계로 보여 주었다. 끌점주위의 구역은 그 끌점들에서 선을 쉽게 련결하기 위하여 확장한다. 인력마당의 원구역에서 선택되는 위치들은 그 구역의 끌점에 당겨 진다. 인력마당의 크기는 위치지정을 할수 있게 충분히 크면서도 다른 선들과 될수록 적게 겹치도록 선택한다. 많은 선들이 현시되면 인력구역들이 겹칠수 있고 점들을 정확히 지적하는것이 힘들수 있다. 일반적으로 인력마당의 경계는 현시하지 않는다.

고무줄방법

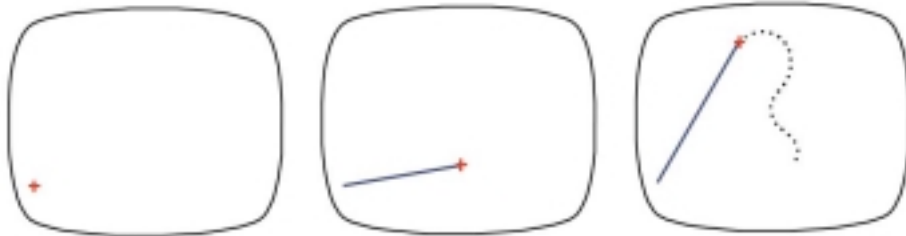
직선은 화면유표을 움직일 때 시작위치로부터 선을 잡아 당기는 고무줄방법을 리용하여 만들고 위치를 정할수 있다. 그림 8-10은 고무줄방법을 보여 주었다. 먼저 선의 한끌점에 대한 화면위치를 선택한다. 다음에 유표를 그 주위에서 움직이면 선이 시작위치로부터 유표의 현재위치까지 현시된다. 두번째 점의 최종적인 화면위치를 선택하면 선의 다른 끌점이 설정된다.

직선의 첫번째 끌점을 선택한다. 유표를 움직이면 초기점으로부터 직선이 나타난다. 두번째 끌점이 선택될 때까지 선은 유표위치를 따라 간다.

그림 8-10. 선분의 위치를 지정하고 긋는 고무줄방법

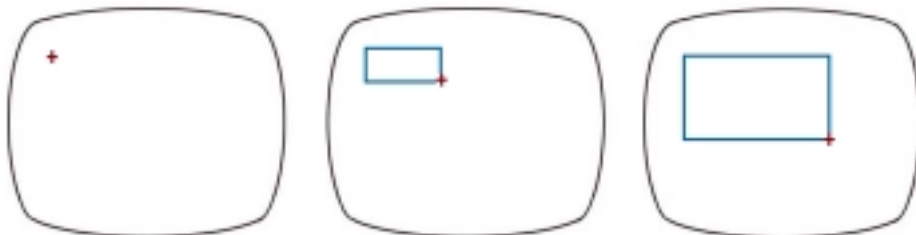
고무줄방법은 직선외에 다른 물체들을 만들고 위치를 정하는데도 리용한다. 그림 8-11은 직4각형의 고무줄방법만들기를 보여 주며 그림 8-12는 고무줄방법에 의한 원생성을 보여 준다.

직4각형의 한쪽 모서리의 위치를 선택한다 유표를 움직이면 직4각형이 나타난다 직4각형의 반대쪽 구석의 위치를 최종적으로 선택한다

그림 8-11. 고무줄방법에 의한 직4각형만들기

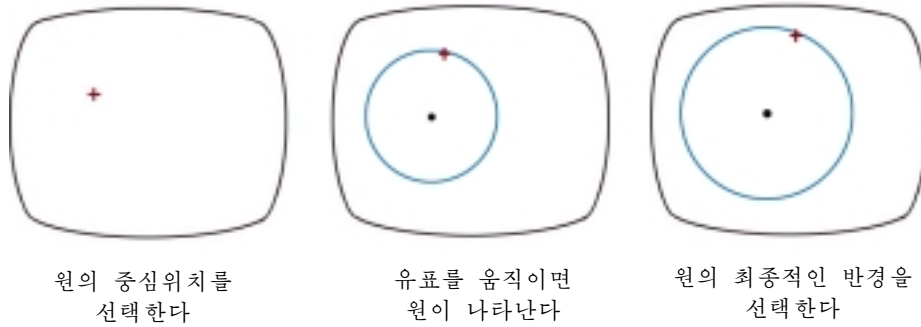



그림 8-12. 고무줄방법에 의한 원만들기

끌기

대화식그림생성에서 흔히 리용하는 한가지 기술은 물체를 화면유표에 의해 끌고 다니는 방법으로 물체들의 위치를 움직이는것이다. 먼저 물체를 선택하고 다음에 유표를 물체를 움직이려는 방향으로 움직인다. 그러면 선택된 물체는 유표를 따라 간다. 물체를 장면의 여러 위치로 끌고 다니는것은 최종위치를 선택하기전에 이러저러한 가능성들을 연구하는데 편리하다.

그림그리기 및 작도

스케치, 작도, 그림그리기에 대한 선택항목들은 여러가지 형식으로 준다. 직선, 다각형, 원들은 앞에서 설명한 방법들로 얻을수 있다. 곡선긋기의 선택항목들은 원호 또는 스플라인과 같은 표준적인 곡선형태들을 리용하거나 손동작스케치절차로 주어 진다. 스플라인곡선은 그 곡선의 대체적인 형태를 주는 불련속화면점들의 모임을 지적하는 방법으로 대화적으로 만든다. 체계는 점들의 이 모임을 다항식곡선에 맞춘다. 손동작으로 만들어 지는 곡선은 도형입력판우에서의 펜의 경로나 현시장치우에서의 화면유표의 경로를 따라가는 방법으로 발생시킨다. 곡선이 현시되면 설계자는 곡선경로를 따라 선택된 점들의 위치를 조절하여 곡선의 형태를 변경시킬수 있다.

일반적으로 그림그리기 및 작도프로그람들은 선의 너비, 선의 형태, 기타 속성선택항목들도 가지고 있다. 이 선택항목들은 4장에서 설명한 방법으로 실현한다. 여러가지 붓형태, 붓무늬, 색조합, 물체의 형태, 면의 걸무늬들은 미술가의 워크스테이션을 비롯한 많은 체계들에서 사용한다. 일부 그림그리기체계들은 펜에 대한 미술가의 손의 압력에 따라 선의 너비와 붓의 획을 변화시킨다. 그림 8-13에는 미술가가 여러가지 물체모양과 각이한 겉면무늬, 장면에 대한 여러가지 조명조건들을 선택할수 있게 한 그림그리기프로그람에서 리용되는 창문과 차림표체계를 보여 주었다.

그림 8-13. 미술가의 그림그리기프로그람에 대한 대면부의 한 형태

6절. 가상현실환경

대표적인 가상현실환경을 그림 8-14에 보여 주었다. 이 환경에서 대화식입력은 가상장면에 현시된 물체를 쥐고 움직일수 있는 자료장갑(2장 5절)에 의하여 수행된다. 콤퓨터로 만들어 진 장면은 모자에 설치된 보기체계를 통하여(2장 1절) 립체투영으로 현시된다. 추적장치들은 장면안의 물체위치에 대한 모자와 자료장갑의 위치와 방향을 계산한다. 이 체계에서 사용자는 장면을 따라 움직일수 있으며 자료장갑에 의하여 물체의 위치들을 재정렬할수 있다.

가상장면을 발생시키는 다른 방법은 라스터현시장치에 두개의 립체보임상의 립체투영을 엇바꾸어 현시하는것이다. 이때 장면은 립체안경을 통하여 본다. 대화식물체처리는 또한 자료장갑 그리고 장면안의 물체들의 위치에 대한 장갑의 위치와 방향을 감시하는 추적장치에 의해서도 수행할수 있다.

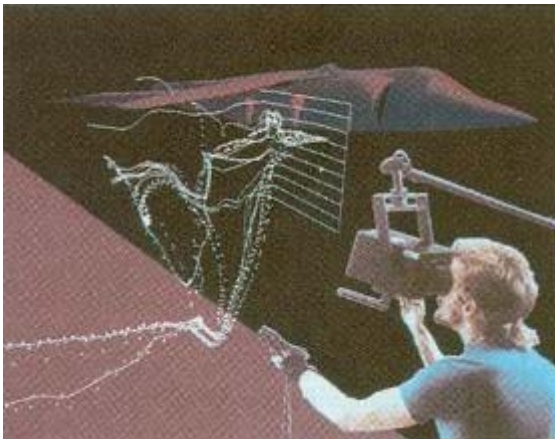



그림 8-14. BOOM(Fake Space Labs, Inc.)이라고 하는 머리추적립체현시장치와 자료장갑(VPL, Inc.)을 리용하여 연구자는 Harrier분사식비행기주위에서의 불안정한 흐름에 대한 시험을 대화적으로 처리하고 있다.

요약

응용프로그람들에서 대화는 응용프로그람의 함수들을 설명하는 사용자모형으로부터 설계된다. 대화의 모든 구성요소들은 응용분야의 언어로 주어 진다. 전기설계, 건축설계프로그람들이 그 실례이다.

도형대면부는 일반적으로 창문들과 그림기호들을 리용하여 설계한다. 창문체계는 사용자들이 창문을 열고, 닫고, 위치를 재설정하고 크기를 변경시킬수 있는 차림표들과 그림기호들을 가진 창문관리대면부를 준다. 창문체계는 여러가지 도형처리연산은 물론 이런 연산들을 수행할수 있는 루틴들을 가지고 있다. 일반적인 창문체계들은 다중창문관리자를 가지도록 설계된다. 그림기호는 응용처리나 조종처리의 식별을 빨리 할수 있도록 설계된 도형적인 기호들이다.

사용자대면부설계에서 고려할 문제는 사용하기 쉽고 명백하고 적응성이 있게 하는것이다. 특히 도형대면부는 사용자가 대화과정에서 일관성을 유지할수 있게 하며 각이한 수준의 사용자들이 사용할수 있게 설계하여야 한다. 보충적으로 대면부는 사용자가 기억해 두어야 하는 내용을 최소로 하며 충분한 반결합이 보장되고 적당한 되돌아 가기가 진행되며 오유처리를 할수 있게 설계되여야 한다.

도형처리프로그람들에 대한 자료입력은 여러가지 각이한 하드웨어장치들로부터 진행될수 있는데 같은 부류의 자료를 입력하는데 하나이상의 장치들이 쓰일수 있다. 입력장치들에 대한 론리적분류를 도입함으로써 도형입력함수들을 매개 입력하드웨어와 독립적으로 설계할수 있게 되였다. 즉 장치들이

마우스나 도형입력판과 같은 하드웨어이름이 아니라 입력자료의 형에 따라 분류된다. 위치지정장치, 획긋기장치, 문자렬장치, 수값장치, 선택장치, 잡기장치의 6가지 론리장치들이 일반적으로 사용된다. 위치지정장치는 프로그람에서 하나의 자리표위치를 입력하는데 쓰이는 장치이다. 획긋기장치는 자리표들의 흐름을 입력한다. 문자렬장치는 본문을 입력하는데 쓰인다. 수값장치는 모든 크기값을 입력하는데 쓰이는 입력장치이다. 선택장치는 차림표를 선택한다. 그리고 잡기장치는 구조물을 포착한다.

입력함수들은 도형처리프로그람들에서 세가지 입력방식으로 정의된다. 요청방식은 응용프로그람의 조종하에서 입력이 주어 진다. 표본방식은 입력장치와 프로그람이 결합되여 동작한다. 사건방식은 입력장치가 자료넣기를 시작하고 자료처리를 조종한다. 일단 론리적분류가 되고 해당한 부류의 자료입력에 사용될 개별적인 론리적장치에 대하여 방식이 선택되면 프로그람에서 입력함수들이 프로그람에 자료값들을 입력하는데 사용된다.

응용프로그람에서는 서로 다른 방식으로 동작하는 여러개의 물리적입력장치들을 동시에 사용할수 있다. 대화식그림생성방법들은 일반적으로 설계 및 그림그리기프로그람들을 비롯하여 응용의 한가지 변종으로 쓰인다. 이 방법들은 사용자에게 물체의 위치지정, 미리정의된 방향이나 기준에 대한 맞추기, 그림의 스케치, 화면에서 물체의 끌고 다니기를 할수 있게 해준다. 격자, 인력마당 그리고 고무줄방법들은 위치지정이나 기타 그림그리기조작을 돕는데 쓰인다.

참고문헌

사용자대면부설계에 대한 요점은 Apple(1987), Bleser(1988), Digital(1989), OSF/ MOTIF(1989)에 있다. X Window System에 대한 정보는 Young(1990), Cutler, Gilly, Reilly(1992)를 보시오. 대면부설계에 대한 보충적인 설명은 Phillips(1977), Goodman과 Spence(1978), Lodding(1983), Swezey와 Davis(1983), Carroll과 Carrithes(1984), Foley, Wallace 그리고 Chan(1984), Good(1984)에서 찾아 볼수 있다.

론리적(혹은 가상적)입력장치개념의 발생발전은 Wallace(1976)과 Resenthal(1982) 등에서 설명하였다. 입력장치분류에 대한 최초의 설명은 Newman(1968)에서 찾아 볼수 있다.

PHIGS에서의 입력연산들은 Hopgood와 Duce(1991), Howard(1991), Gaskins(1992), Blake(1993)에서 찾아 볼수 있다. GKS입력함수들에 대한 정보는 Hopgood(1983)들과 Enderle, Kansg, Pfaff(1984)를 보시오.

련습문제

8-1. 자신이 잘 알고 있는 도형처리의 몇가지 응용분야를 선택하고 그 분야의 도형처리응용에서 사용자대면부설계의 기초로 될수 있는 사용자모형을 만드시오.

8-2. 사용자대면부에 줄수 있는 가능한 방조편의를 적으시오. 그리고 각이한 수준의 사용자들을 대상할수 있는가를 검토하시오.

8-3. 되돌아가기와 오유를 처리하는 가능한 방도를 요약하시오. 그리고 어느 방법이 초학자들에게 더 적당하며 어느것이 경험자들에게 더 적당한가를 말하시오.

8-4. 사용자에게 차림표를 제시하는 가능한 형태를 적으시오. 그리고 그것들이 각각 어떤 환경에서 적당한가를 설명하시오.

8-5. 각이한 수준의 사용자들에 의한 반결합쌍에 대하여 고찰하시오.

8-6. 다중겹침창문을 가지는 화면설계를 다루는데서 창문관리자가 수행해야 할 함수들을 적으시오.

8-7. 창문관리자프로그람을 설계하시오.

8-8. 그림그리기프로그람에 대한 사용자대면부를 설계하시오.

8-9. 2준위계층모형화프로그람에 대한 사용자대면부를 설계하시오.

8-10. 자신이 잘 아는 임의의 부문에 대하여 그 부문의 모든 사용자를 대상할수 있는 도형처리프로그람에 대한 완전한 사용자대면부를 설계하시오.

8-11. 위치지정장치를 써서 화면에서 물체의 위치를 지정할수 있는 프로그람을 개발하시오. 기하학적모양의 물체들의 차림표가 물체를 선택하고 위치를 지적하는 사용자에게 주어 져야 한다. 이 프로그람은 《작업끝》신호가 주어 질 때까지 임의의 개수의 물체에 대하여 위치를 지적할수 있어야 한다.

8-12. 앞의 련습프로그람을 확장하여 선택된 물체에 대하여 위치를 정하기전에 비례변환하고 회전시킬수 있게 하시오. 변환의 선택과 변환파라메터들은 차림표선택항목으로서 사용자에게 주어 져야 한다.

8-13. 획긋기장치를 써서 사용자가 대화적으로 그림을 스케치할수 있는 프로그람을 쓰시오.

8-14. 입력된 문자형태를 형태서고에 기억시킨것과 맞추어 보는 형태인식절차에 쓰일 방법들을 고찰하시오.

8-15. 눈금이 새겨 진 직선과 미끄럼띠를 화면우에 현시하고 눈금직선을 따라 미끄럼띠를 움직여서 수값을 선택할수 있게 하는 루틴을 쓰시오. 선택되는 수값은 눈금직선가까이에 현시되는 네모꼴안에 반영되여야 한다.

8-16. 눈금이 새겨 진 원과 원을 따라 지적된 각도(°)로써 움직일수 있는 지시기나 미끄럼표식을 현시하는 루틴을 쓰시오. 각도값은 눈금원가까이에 현시되는 작은 직4각형안에 반영되여야 한다.

8-17. 사용자가 지적하는 끝점들사이를 련결시키는 선분들의 모임으로 그림을 만들수 있게 하는 프로그람을 쓰시오. 개별적인 선분들의 자리표들은 위치지정장치로 선택한다.

8-18. 지적되는 끝점들사이를 련결시키는 선분들로 그림을 만들수 있게 하는 작도프로그람을 쓰시오. 새로운 선을 이미 존재하는 선에 쉽게 련결시킬수 있게 그림에서 매개 선주위에 인력마당을 설치하시오.

8-19. 앞의 련습문제의 작도프로그람을 수정하여 선들이 오직 수평 또는 수직으로만 만들어 지게 하시오.

8-20. 격자를 선택항목으로 현시함으로써 선택되는 화면위치들이 격자사귐점에 둥그리기 되도록 할수 있는 작도프로그람을 개발하시오. 프로그람은 위치지정장치로 선택된 선끝점들을 가지고 선긋기를 할수 있는 능력을 가져야 한다.

8-21. 설계자가 고무줄방법으로 직선을 스케치하여 그림을 만들수 있는 루틴을 쓰시오.

8-22. 고무줄방법으로 직선, 직4각형, 원을 만들수 있는 작도프로그람을 쓰시오.

8-23. 기본무늬차림표로부터 선택되는 매개 무늬를 잡기장치에 의하여 해당한 위치에 끌어 가는 방법으로 사용자가 그림을 설계할수 있는 프로그람을 쓰시오.

8-24. 요청방식의 입력함수를 실현하시오.

8-25. 표본방식의 입력함수를 실현하시오.

8-26. 사건방식의 입력함수를 실현하시오.

8-27. 일반적인 요청방식, 표본방식, 사건방식의 입력함수들을 실현하시오.

9장. 3차원개념

3차원장면을 모형화하고 현시하는 경우에는 3차원자리표값들을 사용하는것과 함께 고려하여야 할 여러가지 문제들이 있다. 물체의 외면은 평면과 곡면들의 각이한 조합으로 만들수 있다. 때로 물체내부에 대한 정보도 요구된다. 도형처리프로그람들은 흔히 립체물체의 내부 또는 자름면상을 현시하는 루틴들을 가지고 있다. 또한 일부 기하학적변환들은 2차원보다 3차원공간과 더 많이 관계된다. 실례로 3차원공간에서는 임의의 방향을 가지는 축주위로 물체를 회전시킬수 있다. 그러나 2차원에서는 항상 x y평면에 수직인 축주위로만 회전시킨다. 3차원장면을 현시장치에 넘길 때에는 선택해야 할 파라메터들이 많으므로 3차원의 보기변환은 훨씬 더 복잡하다. 장면의 서술은 보기자리표변환과 3차원보기자리표를 2차원장치자리표로 변환하는 투영루틴을 통해 처리하여야 한다. 선택된 보임상에 대하여서는 장면의 보이는 부분들을 식별하여야 하며 장면에 현실감을 주려면 면실감처리알고리듬들을 적용하여야 한다.

1절. 3차원현시방법

세계자리표로 모형화된 3차원장면을 현시하려면 먼저 《카메라》에 대한 자리표계를 설정하여야 한다. 이 자리표계는 카메라필림면의 위치와 방향을 정의한다(그림 9-1). 이 필림면은 장면안의 물체들의 보임상을 현시하기 위한 평면이다. 그다음 물체서술을 카메라자리표계의 자리표로 넘기고 선택된 현시면에 투영한다. 그러면 물체들을 그림 9-2에서와 같은 선그물구조(륜곽선)형식으로 현시할수도 있고 보이는 면들에 명암을 나타내기 위하여 조명 및 면실감처리기술을 적용할수도 있다.

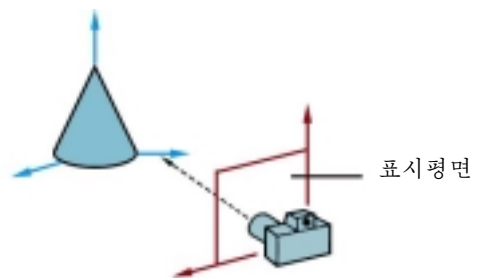



그림 9-1. 3차원장면의 개별적인 보임상을 얻기 위한 자리표계

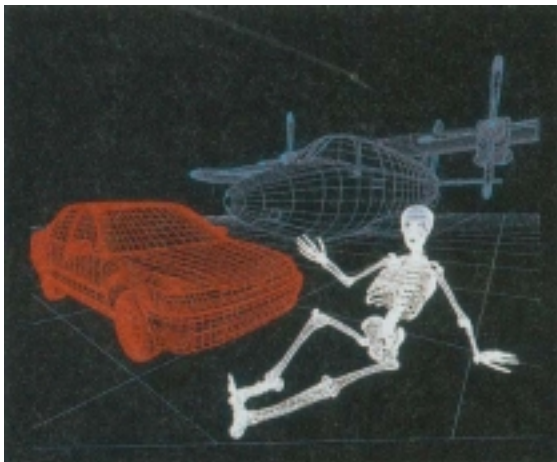



그림 9-2. 물체형태들에 대한 상품화된 자료기지에서 꺼낸 보이지 않는 선을 제거한 세개 물체의 선그물구조현시(자료기지안에서 매개 물체는 자리표점들의 격자로 정의되는데 그것은 선그물구조형식 또는 면실감처리된 형식으로 볼수 있다.)

평행투영

립체물체의 보임상을 만드는 한가지 방법은 물체면우의 점들을 평행선들을 따라 현시면에 투영하는것이다. 서로 다른 보기위치들을 선택하고 물체우의 보이는 점들을 현시면에 투영하면 그림 9-3

에서와 같이 물체의 서로 다른 2차원보임상이 얻어 진다. 평행투영에서는 세계자리표장면안의 평행선들이 2차원현시면에 평행선으로 투영된다. 이 수법은 공학 및 건축그림들에서 물체의 상대적인 비례들이 유지되는 보임상들의 모임으로 물체를 표현하는데 리용된다. 이때 립체물체의 형태는 기본적인 보임상들로부터 다시 생성시킬수 있다.

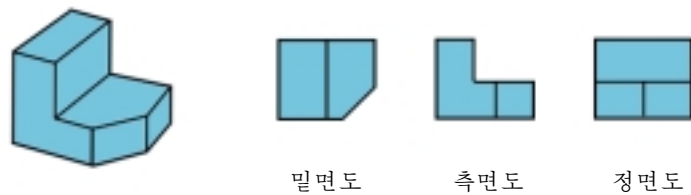



그림 9-3. 여러가지 보기위치에서의 상대적인 비례관계를 보여 주는 물체의 3개의 평행투영상

원근투영

3차원장면의 보임상을 만드는 다른 방법은 점들을 한 점에 집중되는 경로를 따라 현시면에 투영하는것이다. 이것은 보기위치로부터 먼 물체는 보기위치에 가까운 동일한 크기의 물체보다 더 작게 현시되게 한다. 원근투영에서는 장면안의 평행선들이 현시면에서 평행이 아닌 집중되는 선으로 투영된다. 원근투영을 리용하면 현시된 장면들이 보다 현실적으로 보인다. 왜냐하면 그것은 우리의 눈이나 카메라렌즈가 화상을 형성하는것과 같기때문이다. 그림 9-4에 보여 준 원근투영보임상에서 평행선들이 배경의 먼 점에서 보이는 선들로 나타나 있으며 보기위치로부터 먼 물체들은 가까운 물체보다 더 작게 나타나 있다.

그림 9-4. 비행장장면의 원근투영보임상

깊이삽입

약간의 례외는 있지만 매 보기방향에서 어느것이 현시되는 물체의 앞이고 어느것이 뒤인가를 쉽게 식별할수 있게 하는데서 깊이정보가 중요하다. 그림 9-5는 선그물구조의 물체가 깊이정보없이 현시되는 경우에 있을수 있는 모호함을 설명한다. 립체물체의 2차원표현에서 깊이정보를 주는데는 여러가지 방법들이 있다.

선그물구조현시에서 깊이를 나타내는 간단한 방법은 물체들의 세기를 보기위치로부터 그것들의 거리에 따라 변화시키는것이다. 그림 9-6은 깊이삽입으로 현시한 선그물구조물체를 보여 준다. 보기위치에 제일 가까운 선들은 제일 밝은 세기로 현시되고 멀리 떨어 져 있는 선들은 약한 세기로 현시되여 있다. 깊이삽입은 최대 및 최소세기(색)값과 세기들이 변화될 거리들의 범위를 선택하여 적용한다.

깊이삽입의 다른 응용은 대기효과를 물체들에 대하여 느껴 지는 세기로 모형화하는것이다. 먼지립자, 안개, 연기에 의한 빛산란때문에 멀리 있는 물체들은 가까이에 있는 물체보다 희미하게 나타난다. 일부 대기효과들은 물체에 대하여 느껴 지는 색을 변화시킬수 있는데 이런 효과들은 깊이삽입으로 모형화할수 있다.

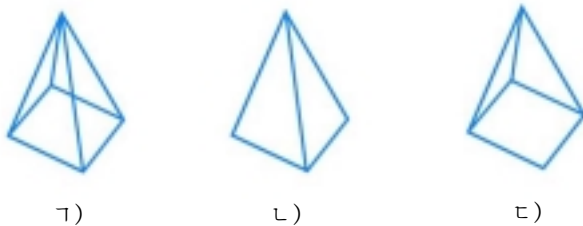



그림 9-5. ㄱ의 각추의 선그물구조표현은 ㄴ처럼 보기방향이 정점에서 아래로 향한것인지 또는 ㄷ처럼 아래바닥으로부터 웃쪽으로 향한것인지를 알수 있는 깊이정보를 가지고 있지 않다.

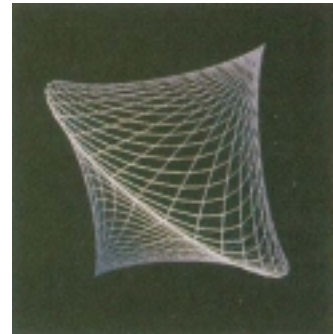



그림 9-6. 깊이삽입으로 현시된 선그물구조의 물체선들의 세기가 물체의 앞에서부터 뒤로 가면서 감소된다.

보이는 선 및 면의 식별

어떤 방식으로 보이는 선들을 식별해 내면 선그물구조현시에서 깊이관계도 명백하게 할수 있다. 제일 간단한 방법은 보이는 선들을 밝기강조하거나 서로 다른 색으로 현시하는것이다. 공학그림들에서 일반적으로 리용하는 수법은 보이지 않는 선들을 파선으로 현시하는것이다. 다른 방법은 그림 9-5 ㄴ와 ㄷ에서와 같이 보이지 않는 선들을 단순히 제거해 버리는것이다. 그러나 보이지 않는 선을 제거하면 또한 물체의 뒤면들의 형태에 대한 정보도 제거된다. 이 보이는 선 식별방법들은 또한 물체의 보이는 면들도 식별할수 있게 한다.

물체가 색이나 명암을 가진 면으로 현시되는 경우에는 보이지 않는 면들이 감추어 지도록 보이는 면들에 면실감처리절차들을 적용한다. 보이는 면알고리듬들가운데서 어떤것들은 보기평면을 지나가는 매 화소마다에 보임성을 설정한다. 또 다른 알고리듬들은 물체의 면에 대하여 전체로서 보임성을 결정한다.

면실감처리

현시에서 현실감은 장면안에서의 조명조건과 면의 특성에 따라 물체들의 면의 세기를 설정하는 방법으로 나타낸다. 조명에는 광원의 세기와 위치 및 장면에 요구되는 일반적인 배경조명이 들어 간

다. 물체들의 면의 특성에는 투명도와 면이 얼마나 거친가 또는 매끈한가가 포함된다. 그다음 장면에 대한 정확한 조명과 그림자구역을 발생시키는데 절차들이 적용되게 된다. 그림 9-7에 현시된 장면에서는 현실감을 내기 위하여 면실감처리방법을 원근 및 보이는 면식별과 결합하고 있다.

그림 9-7. 통계적광선추적법으로 만들어 낸 현실감이 있는 방(여기에서는 원근투영, 면의 걸문양입히기, 조명모형을 적용하고 있다.)

분해된 보임상과 잘라 낸 보임상

많은 도형처리프로그람들에서는 물체들에 대하여 내부의 세부들을 기억시킬수 있는 계층적인 구조로 정의한다. 이때 이런 물체들의 분해된 상과 잘라 낸 상은 내부구조와 물체부분들사이의 관계를 보여 주는데 리용된다. 그림 9-8은 기계설계에서의 몇가지 분해된 상들을 보여 준다. 물체를 잘라 보는것은 그의 구성부분들로 분해하여 보는것과 다르다. 이것은 내부구조를 보여 주기 위하여 보이는 면 부분들을 제거한다.

ㄱ) ㄴ) ㄷ) ㄹ)

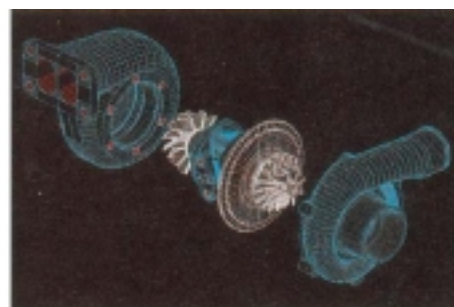
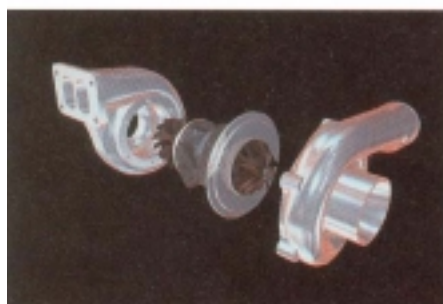



그림 9-8. 완전히 실감처리되고 조립된 화면현시(ㄱ)를 분해된 선그물구조현시(ㄴ), 면실감처리된 분해된 현시(ㄷ), 면실감처리되고 색부호화된 분해된 현시(ㄹ)로도 볼수 있다.

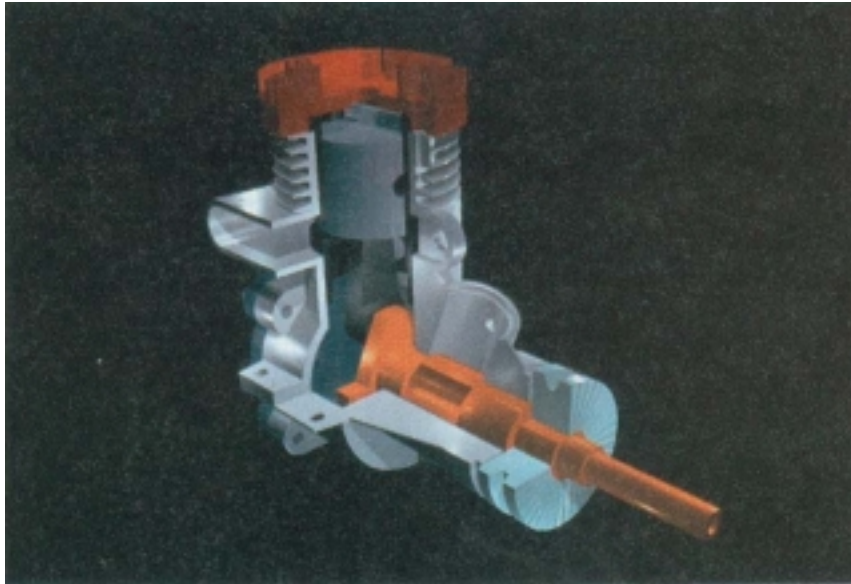



그림 9-9. 내부구성부분들의 구조와 판계를 보여 주는 잔디깍는 기계기관의 색부호화된 잘라 낸 보임상

3 차원 및 립체보기

콤퓨터로 만든 장면에 현실감을 부여하는 다른 방법은 3차원 또는 립체상을 리용하여 물체를 현시하는것이다. 2장에서 본바와 같이 3차원상은 라스터화상을 진동식탄성거울로부터 반사시켜 얻을수 있다. 거울의 진동은 CRT에서의 장면현시와 동기된다. 거울이 진동하면 장면안의 매개 점이 그의 깊이에 대응하는 위치에 투영되도록 초점거리가 변한다.

립체장치들은 왼쪽 눈과 오른쪽 눈에 각각 하나씩 장면의 두개의 보임상을 준다. 이 두 보임상은 보는 사람의 두 눈위치에 대응하는 보기위치들에서 발생시킨다. 이 두 보임상은 라스터현시장치에서 엇갈리는 재생주기로 현시되는데 현시장치의 재생주기와 동기를 맞추어 처음에는 한쪽 렌즈를 다음에는 다른쪽 렌즈를 번갈아 어둡게 하는 안경을 통해 보게 된다.

2절. 3차원도형처리프로그람

3차원프로그람들의 설계에서는 2차원프로그람들에서 제기되지 않았던것들도 일부 고찰해야 한다. 두 프로그람들사이의 중요한 차이는 3차원프로그람에서는 장면의 서술을 평탄한 보기면에로 넘기는 방법들을 가지고 있어야 한다는것이다. 각이한 보임상들의 선택과 여러가지 투영기술들의 리용에 대한 실현절차들을 고찰해야 한다. 또한 립체물체들의 면을 어떻게 모형화하며 보이는 면들을 어떻게 식별하며 물체들의 변환을 공간상에서 어떻게 수행하며 3차원에 의하여 도입되는 추가적인 공간적인 특성들을 어떻게 서술하는가 하는것들을 고찰해야 한다. 뒤장들에서 이런 문제들을 구체적으로 고찰한다.

3차원프로그람들에 대한 다른 고찰들은 2차원방법의 간단한 확장이다. 세계자리표서술은 3차원으로 확장되여 사용자에게는 다음과 같이 지적하면 호출되는 출력 및 입력루틴들이 주어 진다.

```
polyline3 (n, wcPoints )
fillarea3 (n, wcPoints )
text3 (wcPoints, string)
getLocator3 ( wcPoint )
translate3 ( translateVector, matrixTranslate )
```

여기서 점과 벡토르는 3개의 성분들에 의해 지적되며 변환행렬은 4개의 행과 4개의 렬을 가진다.

기하학적고찰과는 독립인 2차원속성함수들은 2차원과 3차원응용에 다같이 적용될수 있다. 색, 선의 형, 표식의 속성, 본문폰트에 대하여서는 새로운 속성함수들을 정의하지 않아도 된다. 그러나 문자렬의 방향을 선정하는 속성절차들은 임의의 공간적인 방향을 줄수 있게 확장되여야 한다. 웃벡토르와 관련되는 본문속성루틴들은 문자렬에 임의의 공간적인 방향이 주어 질수 있도록 z자리표자료를 포함시켜 확장하여야 한다. 무늬참조점의 위치를 지정하고 무늬를 채우기구역에 넘기는것과 같은 구역채우기루틴들은 채우기구역의 면과 무늬평면의 여러가지 방향을 조절할수 있게 확장되여야 한다. 또한 앞장들에서 설명한 대부분의 2차원구조물조작들이 3차원프로그람에 넘겨 질수 있다.

그림 9-10은 세계자리표장면을 현시하는 3차원변환처리흐름의 일반적인 단계들을 보여 준다. 물체의 정의를 보기자리표로 변환한 다음 주사변환알고리듬들을 적용하여 현시면에 투영되는 라스터화상을 기억시킨다.

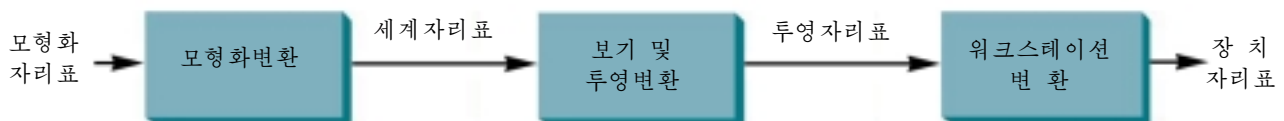



그림 9-10. 세계자리표장면의 보임상을 장치자리표로 변환하는 흐름

10장. 3차원물체의 표현

도형처리장면들에는 나무, 꽃, 구름, 바위, 벽돌, 나무판자, 고무, 종이, 대리석, 강철, 유리, 합성수지, 천 등 각양각색의 수많은 종류의 물체들이 포함될수 있다. 때문에 이런 서로 다른 재료들의 모든 특성이 다 반영되게 물체를 표현할수 있는 통일적인 방법이란 있을수 없다. 장면을 현실감있게 현시하기 위하여서는 물체의 특성을 잘 나타내는 모형화를 리용해야 한다.

다면체 및 타원체와 같은 간단한 유클리드물체들은 다각형과 2차곡면으로 정확히 표현할수 있다. 스플라인곡면과 그의 구성기법은 비행기의 날개, 이바퀴, 곡면을 가지는 기타 공학적인 구조물들의 설계에 쓰인다. 프락탈구성 및 립자계와 같은 수속적인 방법들은 구름, 수풀 기타 자연물체들을 잘 표현할수 있게 한다. 호상작용하는 힘계를 리용하는 물리학적모형화방법들은 천쪼각 또는 묵덩어리와 같은 비강성움직임을 표현하는데 리용할수 있다. 8분나무부호화는 의학CT화상과 같이 물체의 내부특징을 표현하는데 리용한다. 등값면현시, 체적실감처리, 기타 가시화수법들을 3차원불련속자료모임에 적용하면 자료의 시각적인 표현을 얻을수 있다.

물체를 표현하는 방법은 일반적으로 크게 두가지 부류로 나눈다. 그러나 모든 표현들이 이 두 부류중 어느 하나에만 떨어 지는것은 결코 아니다. **경계표현**(boundary representation, B-reps)방법에서는 3차원물체를 경계면들의 모임으로 표현한다. 경계표현의 대표적인 실례는 다각형쪼각표현과 스플라인곡면쪼각표현이다. **공간분할표현**(space-partitioning representation)방법에서는 물체를 포함하는 공간구역을 겹치지 않는 작은 립체(보통 바른6면체)들의 모임으로 련속 세분하는 방법으로 물체의 내부특성을 표현한다. 3차원물체에 대한 일반적인 공간분할표현은 8분나무표현이다. 이 절에서는 여러가지 표현방법들의 특징과 응용에 대하여 고찰한다.

1절. 다각형면

3차원도형처리물체에 대한 가장 일반적인 경계표현은 물체를 둘러 싸는 겉면을 다각형들의 모임으로 표현하는것이다. 많은 도형처리체계들에서는 모든 물체표현을 겉면의 다각형들의 모임으로 기억시킨다. 이것은 모든 면들이 선형방정식으로 표현되기때문에 면실감처리와 물체의 현시를 간단하게 하고 속도를 높인다. 이런 리유로 흔히 다각형표현을 도형처리물체의 《표준표현》이라고 한다. 다각형표현 하나만을 사용할수 있게 한 경우도 일부 있으나 많은 프로그람들은 물체를 스플라인곡면과 같은 다른 방법들로도 표현할수 있게 하고 그것을 후에 다각형표현으로 변환한다.

다면체와 같은 물체에서는 다각형표현이 물체겉면의 특징을 정확히 나타낸다. 그러나 다른 물체들의 겉면에 대하여서는 다각형그물근사를 진행하고 쪽무이 또는 타일붙이기를 진행한다. 그림 10-1에서는 원기둥면을 다각형그물로 표현하였다. 일반적으로 설계응용과 립체모형화응용에서 이러한 표현을 쓴다. 왜냐하면 선그물구조륜곽선은 면의 일반적인 구조를 빨리 볼수 있게 하기때문이다. 현실감을 주는 실감처리는 다각형면들을 건너 가면서 명암무늬들을 보간하여 다각형변들의 경계를 없애거나 줄인다. 그리고 곡면에 대한 다각형그물근사는 곡면을 더 작은 다각형쪼각들로 세분하는 방법으로 개선할수 있다.

그림 10-1. 뒤(보이지 않는)선이 제거된 원기둥의 선그물구조표현

다각형표

다각형면은 정점들의 자리표와 관련속성파라메터들의 모임으로 지적된다. 매개 다각형에 대한 정보가 입력되면 자료를 자료표에 넣고 후에 장면안에서 물체들을 처리, 현시, 조작하는데 리용한다.

다각형자료표는 두개의 표 즉 기하학적자료표와 속성표로 구성할수 있다. 기하학적자료표에는 정점들의 자리표와 다각형면의 공간적인 방향을 식별하는 파라메터들이 포함된다. 속성표에는 물체의 투명도, 면의 반사률, 결문양 등의 파라메터들이 포함된다.

기하학적자료를 편리하게 기억시키기 위하여 세개의 목록 즉 정점표, 변표, 다각형표를 만든다. 물체의 매 정점에 대한 자리표값은 정점표에 기억시킨다. 변표에는 다각형의 정점표에서 매개 변의 정점들을 식별하기 위한 지적자가 포함된다. 그리고 다각형표에는 변표에서 다각형의 매 변들을 식별하기 위한 지적자가 포함된다. 그림 10-2에서는 이 방법을 어떤 물체면의 2개의 린접한 다각형에 대하여 적용하고 있다. 개별적인 물체와 그의 요소다각형면을 쉽게 참조하기 위하여 물체와 요소면의 식별자도 줄수 있다.

정점표	
V_1:	x_1, y_1, z_1
V_2:	x_2, y_2, z_2
V_3:	x_3, y_3, z_3
V_4:	x_4, y_4, z_4
V_5:	x_5, y_5, z_5

변 표	
E_1:	V_1,V_2
E_2:	V_2,V_3
E_3:	V_3,V_1
E_4:	V_3,V_4
E_5:	V_4,V_5
E_6:	V_5,V_1

다각형표	
S_1 :	$E_1, E_2, E_3,$
S_1:	$E_3, E_4, E_5, E_6,$

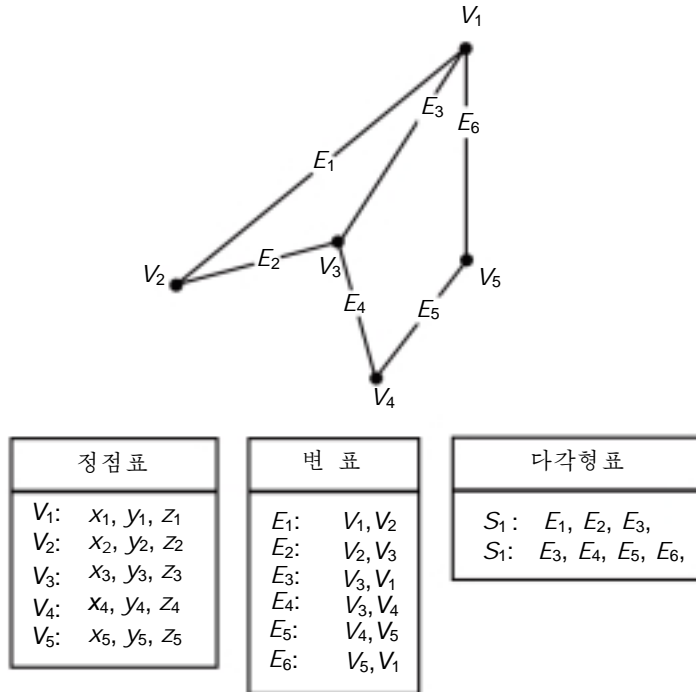



그림 10-2. 6개의 변과 5개의 정점으로 이루어 진 린접한 두 다각형면에 대한 기하학적자료표

그림 10-2에서와 같이 기하학적자료를 세개의 표에 기입하면 매개 물체의 개별적인 요소(정점, 변, 다각형)들을 편리하게 참조할수 있으며 또 변표의 자료들을 리용하여 요소선들을 그음으로써 물체를 효률적으로 현시할수 있다. 다른 방법은 두개의 표 즉 정점표와 다각형표를 리용하는것이다. 그러나 이 방법은 일부 변들이 두번 그려 질수 있으므로 좋지 못하다. 또 다른 방법은 다각형표만을 리용하는것이다. 그러나 이 방법은 매 다각형의 매 정점에 대하여 명백한 자리표값들이 기입되기때문에 자리표정보를 중복시킨다. 또한 변에 대한 정보는 다각형표에 기입된 정점자료로부터 얻어 내야 한다.

E_1:	V_1, V_2, S_1
E_2:	V_2, V_3, S_1
E_3:	V_3, V_1, S_1,S_2
E_4:	V_3, V_4, S_2
E_5:	V_4, V_5, S_2
E_6:	V_5, V_1, S_2

그림 10-3. 다각형표에 대한 지적자를 포함하도록 확장된 그림 10-2의 면에 대한 변표

정보를 더 빨리 얻어 내기 위하여 그림 10-2의 자료표에 보충적인 정보를 추가할수 있다. 실례로 다각형들사이의 공통변을 더 빨리 식별할수 있게 변표를 다각형표에 대한 지적자가 포함되도록 확장할수 있다(그림 10-3). 특히 이것은 변을 따라 한 다각형으로부터 다음 다각형에로 면의 명암을 부드럽게 달리 하여야 하는 실감처리절차에서 편리하다. 류사하게 정점표는 정점들에 대응하는 변들을 교차참조하도록 확장할수 있다.

기하학적자료표에는 또한 매 변의 경사도와 매 다각형의 자리표범위도 추가할수 있다. 정점들을 입력할 때 변의 경사도를 계산할수 있으며 매 다각형의 최소 및 최대 x, y, z값을 식별하기 위하여 자리표값들을 조사할수 있다. 변의 경사도와 다각형의 테두리직4각형정보는 후에 면실감처리와 같은 처리에 필요하다. 자리표범위는 또한 일부 보이는 면결정알고리듬에도 리용된다.

복잡한 물체에서는 기하학적자료표에 정점들과 변들이 많이 포함될수 있기때문에 자료의 일관성과 완전성을 검사하는것이 중요하다. 특히 대화식응용들에서는 정점, 변, 다각형을 지적할 때 물체의 현시를 외곡할수 있는 일정한 입력오유가 일어 날수 있다. 오유검사는 자료표에 정보가 많이 포함될수록 더 쉽다. 따라서 오유검사는 많은 정보를 제공해 주는 세개의 자료표(정점, 변, 다각형)를 리용할 때 더 쉽다. 도형처리프로그람들에서 수행될수 있는 몇가지 검사로는 첫째로 각 정점이 적어도 2개 변의 끝점으로 기입되여 있는가, 둘째로 매개 변이 적어도 한개 다각형의 부분으로 되여 있는가, 셋째로 매 다각형이 닫겨 있는가, 넷째로 매 다각형이 적어도 하나의 공통변을 가지고 있는가, 다섯째로 변표에 다각형에 대한 지적자가 포함될 때 다각형지적자에 의하여 참조되는 매 변이 그 다각형에 대한 지적자를 가지고 있는가이다.

평면의 방정식

3차원물체를 현시하려면 물체에 대한 입력자료표현을 여러가지 절차를 통하여 처리하여야 한다. 이 처리걸음들에는 모형화 및 세계자리표표현의 보기자리표, 나아가서 장치자리표에로의 변환, 보이는 면의 식별, 면실감처리절차의 적용 등이 포함된다. 일부 처리들에서는 물체의 개별적인 요소면의 공간적인 방향에 대한 정보가 필요하다. 이 정보는 정점들의 자리표값과 다각형면을 표현하는 방정식으로부터 얻어 진다.

평면의 방정식은

$$Ax + By + Cz + D = 0 \qquad (10-1)$$

형식으로 표현할수 있다. 여기서 (x, y, z)는 평면상의 임의의 점이며 결수 A, B, C, D는 평면의 공간적인 특성을 나타내는 상수이다. A, B, C, D의 값은 평면에서 같은 직선우에 있지 않는 세개의 점에 대한 자리표값을 리용하여 세개의 평면방정식을 풀어 얻을수 있다. 이를 위하여 다각형의 련속된 세 정점 (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)을 선택하고 비 $A/D, B/D, C/D$에 대하여 다음의 련립선형방정식을 푼다.

$$(A/D)x_k + (B/D)y_k + (C/D)z_k = -1, \qquad k = 1, 2, 3 \qquad (10-2)$$

이 련립방정식의 풀이는 크라메르규칙을 리용하여 행렬식형식으로

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix} \qquad B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix}$$
$$C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} \qquad D = -\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} \qquad (10-3)$$

와 같이 얻을수 있다. 행렬식을 전개하면 평면의 결수들에 대한 계산을

$$\begin{aligned} A &= y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2) \\ B &= z_1(x_2 - x_3) + z_2(x_3 - x_1) + z_3(x_1 - x_2) \\ C &= x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2) \\ D &= -x_1(y_2 z_3 - y_3 z_2) - x_2(y_3 z_1 - y_1 z_3) - x_3(y_1 z_2 - y_2 z_1) \end{aligned} \qquad (10-4)$$

의 형식으로 쓸수 있다. 정점들의 값과 기타 정보가 다각형자료구조에 들어 갈 때 A, B, C, D의 값들이 매 다각형에 대하여 계산되며 다각형의 다른 자료들과 함께 기억된다.

공간에서 평면의 방향은 그림 10-4에 보여 준바와 같이 평면의 법선벡토르로 표현할수 있다. 이 벡토르는 직각자리표성분 (A, B, C)를 가지며 파라메터 A, B, C는 식 10-4에서 계산된 평면의 결수들이다.

보통 물체내부를 둘러 싸는 다각형면을 취급하므로 면의 량쪽을 구별할 필요가 있다. 물체의 안쪽 면을 《내부면》이라고 하며 보이는 즉 바깥쪽 면을 《외부면》이라고 한다. 오른손자리표계에서 평면의 바깥쪽 면을 볼 때 다각형의 정점들이 시계바늘반대방향으로 지적되면 법선벡토르는 내부에서 외부로 향하게 된다. 이것을 그림 10-5의 단위바른6면체의 한개 면에서 보여 준다.

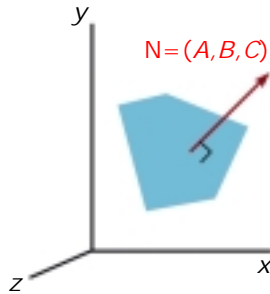



그림 10-4. 방정식 $Ax+By+Cz+D=0$ 에 의하여 표현되는 평면에 수직인 벡토르 **N**은 직각자리표성분 (A,B,C)를 가진다.

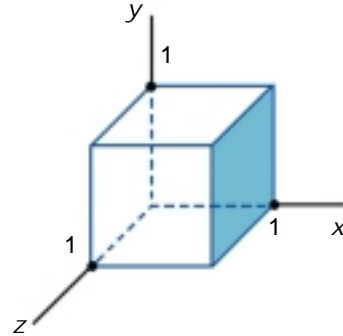



그림 10-5. 단위바른6면체의 색 칠된 다각형면은 평면방정식 $x-1=0$ 과 법선벡토르 $\mathbf{N}=(1, 0, 0)$을 가진다.

그림 10-5에 보여 준 색칠된 면의 법선벡토르의 성분들을 결정하기 위하여 그 다각형의 둘레를 따라 4개의 정점중 3개를 선택한다. 이 점들은 바른6면체의 밖에서 원점쪽으로 볼 때 시계바늘반대방향으로 선택된다. 식 10-4에서 이 정점들의 자리표를 리용하여 평면의 결수 $A=1$, $B=0$, $C=0$, $D=-1$을 얻는다. 따라서 이 평면의 법선벡토르는 정의 x축방향으로 향한다.

평면의 법선벡토르의 성분들은 벡토르적을 계산하여도 얻을수 있다. 오른손직각자리표계에서 면을 외부에서 볼 때 시계바늘반대방향순서로 세개의 정점위치 $\mathbf{V}_1$, $\mathbf{V}_2$, $\mathbf{V}_3$을 선택한다. $\mathbf{V}_1$부터 $\mathbf{V}_2$까지와 $\mathbf{V}_1$부터 $\mathbf{V}_3$까지 두개의 벡토르를 정의하면 벡토르적 **N**은

$$\mathbf{N} = (\mathbf{V}_2 - \mathbf{V}_1) \times (\mathbf{V}_3 - \mathbf{V}_1) \tag{10-5}$$

으로 계산된다. 이것은 평면의 파라메터 A, B, C의 값들을 준다. 다음에 이 값들과 함께 다각형의 정점들중 하나의 자리표를 평면의 방정식 10-1에 대입하고 D에 대하여 풀면 파라메터 D의 값을 얻을수 있다. 법선 **N**과 평면의 임의의 점의 위치 **P**를 리용하면 평면의 방정식은 벡토르형식으로

$$\mathbf{N} \cdot \mathbf{P} = -D \tag{10-6}$$

와 같이 표현할수 있다.

평면의 방정식은 그 평면에 대한 점의 공간적인 위치를 식별하는데 리용된다. 파라메터 A, B, C, D를 가지는 평면에 놓이지 않는 임의의 점 (x, y, z)에 대하여서는

$$Ax + By + Cz + D \neq 0$$

이다. $Ax+By+Cz+D$의 부호(부 또는 정)에 따라 점이 평면의 내부에 있는가 또는 외부에 있는가를 식별할수 있다.

$Ax+By+Cz+D<0$이면 점 (x, y, z)는 면의 내부에 있다.

$Ax+By+Cz+D>0$이면 점 (x, y, z)는 면의 외부에 있다.

이 부등식검사는 오른손직각자리표계에서 면을 외부에서 볼 때 시계바늘반대방향순서로 선택되는 정점들을 리용하여 평면의 파라메터 A, B, C, D를 계산하면 된다. 실례로 그림 10-5에서 색칠된 평면밖의 임의의 점은 부등식 $x-1>0$을 만족시키며 동시에 그 평면안의 임의의 점은 1보다 작은 x자리표값을 가진다.

다각형그물

일부 도형처리프로그람(실례로 PHIGS)들은 물체를 모형화하는 여러가지 다각형함수들을 제공한다. 하나의 평면은 `fillArea`와 같은 함수에 의하여 지적할수 있다. 그러나 물체의 면이 쪽무이로 이루어 질 때에는 면의 요소면들을 그물함수에 의하여 지적하는것이 더 편리하다. 다각형그물의 한가지 형태는 3각형띠이다. 3각형띠함수는 그림 10-6에 보여 준바와 같이 n개 정점들의 자리표가 주어 지면 련결된 n–2개의 3각형들을 만든다. 다른 류사한 함수는 4각형그물이다. 이것은 n×m배렬의 정점들에 대한 자리표가 주어 지면 $(n-1)\times(m-1)$개 4각형들의 그물을 만든다. 그림 10-7에서는 12개의 4각형그물을 만드는 20개의 정점들을 보여 주었다.

그림 10-6. 13 개의 정점들을 련결한 11개의 3각형으로 만들어 진 3각형띠

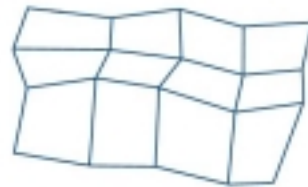



그림 10-7. 5×4 의 입력정점배렬로 만들어 진 12개의 4각형을 가지는 4각형그물

다각형이 세개보다 많은 정점들로 지적될 때에는 모든 정점들이 한 평면에 놓이지 않을수 있다. 이것은 수값오유 또는 정점들에 대한 자리표위치선택에서의 오유때문이다. 이런 경우에 리용되는 한가지 방법은 다각형을 단순히 3각형들로 나누는것이다. 다른 방법은 평면의 파라메터 A, B, C를 근사화하는것이다. 이것은 평균하는 방법으로 할수도 있고 다각형을 자리표평면에 투영하는 방법으로도 할수 있다. 투영방법을 리용할 때 A는 yz평면에서의 다각형의 투영면적에 비례되게, B는 xz평면에서의 투영면적에 비례되게, C는 xy평면에서의 투영면적에 비례되게 취한다.

고급한 도형처리체계들에서는 일반적으로 물체를 다각형그물로 모형화하고 다각형쪼각들을 쉽게 처리하기 위하여 기하학적정보와 속성정보에 대한 자료기지를 설치한다. 면의 걸문양과 특수조명효과의 응용을 비롯하여 초당 몇십몇백만개이상의 명암처리된 다각형(보통3각형)을 현시할수 있는 체계들에는 다각형실감처리를 하드웨어적으로 실현하는 고속장치들이 병합되고 있다.

2절. 곡선과 곡면

3차원의 곡선과 곡면은 물체를 정의하는 수학함수들의 입력모임 또는 사용자가 지적하는 자료점들의 모임에 의하여 현시할수 있다. 함수로 지적될 때 프로그람은 곡선에 대한 정의를 현시면에 투영하고 투영된 함수의 경로를 따라 화소위치들을 현시할수 있다. 곡면에 대한 함수적인 표현은 곡면

을 작은 쪼각들로 나누어 다각형그물로 근사시킨다. 일반적으로 임의의 다각형의 모든 정점들이 그 면에 놓인다는것을 보증하기 위하여 곡면을 3각형쪼각들로 나눈다. 4개 또는 그이상의 정점들에 의하여 지적되는 다각형은 모든 정점들이 한 평면에 놓이지 않을수 있다. 2차곡면과 초2차곡면은 함수표현으로 만들어 지는 곡면들의 실례이다.

불련속자리표점들의 모임으로 물체의 형태를 지적할 때에는 응용의 조건에 따라 지적된 점들을 가장 잘 맞추는 함수표현을 얻게 된다. 스플라인표현이 이 부류의 곡선 및 곡면의 실례이다. 이 방법은 일반적으로 새로운 물체형태를 설계하고 그림을 수자화하며 동화경로를 표현하는데 리용된다. 곡선맞추기방법은 또한 최소두제곱법과 같은 회귀수법들을 리용하여 지적된 곡선함수를 불련속자료모임에 맞춤으로써 자료값들의 그라프를 현시하는데도 리용한다.

곡선 및 곡면의 방정식은 보조변수 또는 비보조변수형식으로 표현할수 있다. 부록에 보조변수 및 비보조변수방정식들에 대한 요약과 비교를 주었다. 콤퓨터도형처리응용에서는 보조변수표현이 일반적으로 더 편리하다.

3절. 2차곡면

2차곡면은 흔히 리용되는 물체로서 그것은 2차방정식으로 표현된다. 2차곡면에는 구, 타원체, 원환체, 포물면, 쌍곡면이 있다. 2차곡면 특히 구와 타원체는 도형처리장면들의 공통적인 요소이며 도형처리프로그람들은 그것을 기초요소로 리용하여 복잡한 물체를 만든다.

구

직각자리표계에서 반경이 r이고 중심이 자리표원점에 있는 구면은 방정식

$$x^2 + y^2 + z^2 = r^2 \qquad (10-7)$$

을 만족시키는 점 (x, y, z)들의 모임으로 정의된다. 또한 구면을 위도각과 경도각을 리용하여 보조변수형식으로 표현할수 있다(그림 10-8).

$$\begin{aligned} x &= r\cos\phi\cos\theta, \qquad -\pi/2 \le \phi \le \pi/2 \\ y &= r\cos\phi\sin\theta, \qquad -\pi \le \theta \le \pi \\ z &= r\sin\phi \end{aligned} \qquad (10-8)$$

식 10-8의 보조변수표현에서는 각 θ와 ϕ에 대하여 대칭범위를 주고 있다. 또한 보조변수방정식을 표준구면자리표를 리용하여 쓸수도 있다. 여기서 각 ϕ는 여위도각(그림 10-9)으로 지적된다. 이때에는 ϕ가 $0 \le \phi \le \pi$범위에서 정의되며 θ는 흔히 $0 \le \theta \le 2\pi$범위에서 취해 진다. 또한 $\phi = \pi u$, $\theta = 2\pi v$로 치환하여 0~1사이 범위에서 정의되는 파라메터 u와 v를 리용하는 표현을 설정할수도 있다.

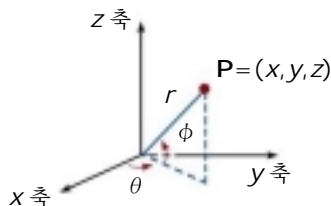



그림 10-8. 반경이 r인 구면에서 보조변수자리표 위치 (r, θ, ϕ)

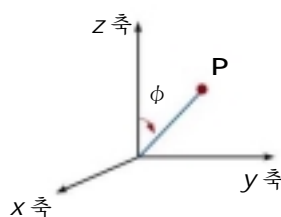



그림 10-9. 각 ϕ에 대하여 여위도각을 리용하는 구면자리표 파라메터 (r, θ, ϕ)

타원체

타원체는 서로 수직인 세개의 방향에서 반경들이 서로 다른 값을 가지는 구면의 확장으로 표현할수 있다(그림 10-10). 중심이 원점에 있는 타원면우의 점에 대한 직각자리표표현은

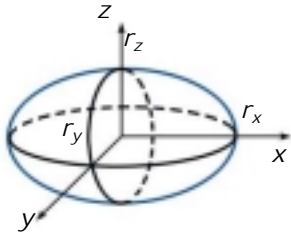



그림 10-10. 반경이 r_x, r_y, r_z이고 중심이 자리표원점에 있는 타원체

$$\left(\frac{x}{r_x}\right)^2+\left(\frac{y}{r_y}\right)^2+\left(\frac{z}{r_z}\right)^2=1 \qquad (10-9)$$

이다. 그리고 타원체에 대한 보조변수표현은 그림 10-8의 위도각 ϕ와 경도각 θ에 의하여

$$\begin{aligned} x &= r_x \cos\phi\cos\theta, && -\pi/2 \le \phi \le \pi/2 \\ y &= r_y \cos\phi\sin\theta, && -\pi \le \theta \le \pi \\ z &= r_z \sin\phi \end{aligned} \qquad (10-10)$$

이다.

원환체

원환체는 그림 10-11에 보여 준바와 같이 가락지모양의 물체이다. 이것은 원 또는 다른 원추곡선을 지적된 축에 대하여 회전시켜 얻을수 있다. 원환체면우의 점에 대한 직각자리표표현은

$$\left[r-\sqrt{\left(\frac{x}{r_x}\right)^2+\left(\frac{y}{r_y}\right)^2}\right]^2+\left(\frac{z}{r_z}\right)^2=1 \qquad (10-11)$$

형식으로 쓸수 있다. 여기서 r는 임의로 주어 진 편위값이다. 원환체에 대한 보조변수표현은 각 ϕ가 360°로 확장된다는것을 제외하면 타원체와 류사하다. 위도 및 경도각 ϕ와 θ를 리용하면 원환체면을

$$\begin{aligned} x &= r_x (r+\cos\phi)\cos\theta, && -\pi \le \phi \le \pi \\ y &= r_y (r+\cos\phi)\sin\theta, && -\pi \le \theta \le \pi \\ z &= r_z \sin\phi \end{aligned} \qquad (10-12)$$

을 만족시키는 점들의 모임으로 표현할수 있다.

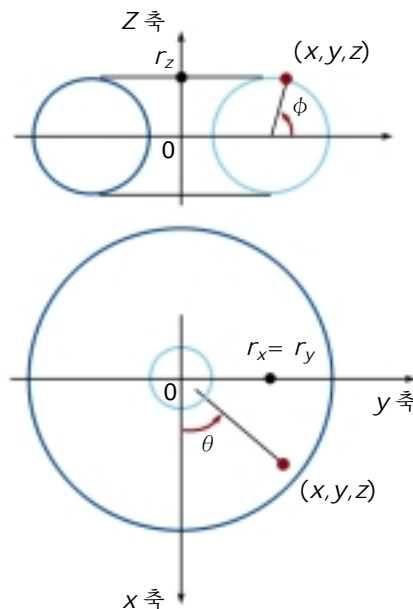



그림 10-11. 자름면이 원형이고 중심이 자리표원점에 있는 원환체

4절. 초2차곡면

초2차곡면은 2차곡면표현의 일반화이다. **초2차곡면**(superquadrics)은 물체의 형태를 보다 신축성 있게 조정하기 위하여 2차곡면의 방정식에 추가적인 파라메터들을 병합하여 얻는다. 사용되는 추가 파라메터들의 수는 물체의 차원수와 같다. 즉 곡선에서는 파라메터가 하나이며 곡면에서는 파라메터가 두개이다.

초타원

초타원에 대한 직각자리표표현은 대응하는 타원의 방정식에서 x와 y항의 지수를 변화시켜 얻는다. 이렇게 하는 한가지 방법은 초타원의 직각자리표방정식을

$$\left(\frac{x}{r_x}\right)^{2/s} + \left(\frac{y}{r_y}\right)^{2/s} = 1 \qquad (10-13)$$

의 형식으로 쓰는것이다. 여기서 파라메터 s에는 임의의 실수값이 주어 진다. $s=1$일 때에는 타원이 얻어 진다.

식 10-13의 초타원에 대한 보조변수방정식은

$$\begin{aligned} x &= r_x \cos^s \theta, \qquad -\pi \le \theta \le \pi \\ y &= r_y \sin^s \theta \end{aligned} \qquad (10-14)$$

로 표현할수 있다. 그림 10-12에서는 파라메터 s의 서로 다른 값들을 리용하여 만든 여러가지 형태의 초타원들을 보여 주었다.

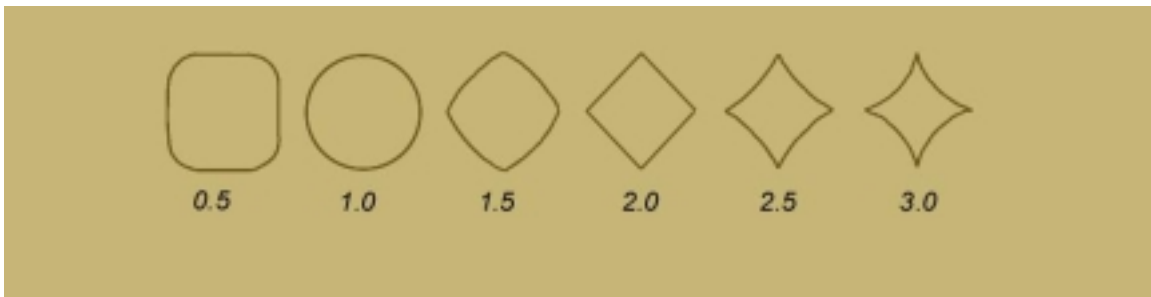



그림 10-12. 파라메터 s의 서로 다른 값과 $r_x=r_y$에 의하여 표시된 초타원

초타원체

초타원체에 대한 직각자리표표현은 타원체에 대한 방정식에 두개의 지수파라메터를 더 포함시켜 얻는다.

$$\left[\left(\frac{x}{r_x}\right)^{2/s_2} + \left(\frac{y}{r_y}\right)^{2/s_2}\right]^{s_2/s_1} + \left(\frac{z}{r_z}\right)^{2/s_1} = 1 \qquad (10-15)$$

$s_1=s_2=1$일 때에는 타원체가 얻어 진다.

식 10-15의 초타원체에 대한 보조변수표현은

$$x = r_x \cos^{s_1}\phi \cos^{s_2}\theta, \qquad -\pi/2 \le \phi \le \pi/2$$
$$y = r_y \cos^{s_1}\phi \sin^{s_2}\theta, \qquad -\pi \le \theta \le \pi \qquad (10-16)$$
$$z = r_z \sin^{s_1}\phi$$

와 같이 쓸수 있다. 그림 10-13에서는 파라메터 s_1와 s_2의 서로 다른 값들을 리용하여 만든 여러가지 형태의 초타원체들을 보여 주었다. 초타원체와 기타 초2차곡면형태들은 가구, 나사를 친 볼트와 같은 복잡한 구조들과 기타 물체들을 만드는데 결합될수 있다.

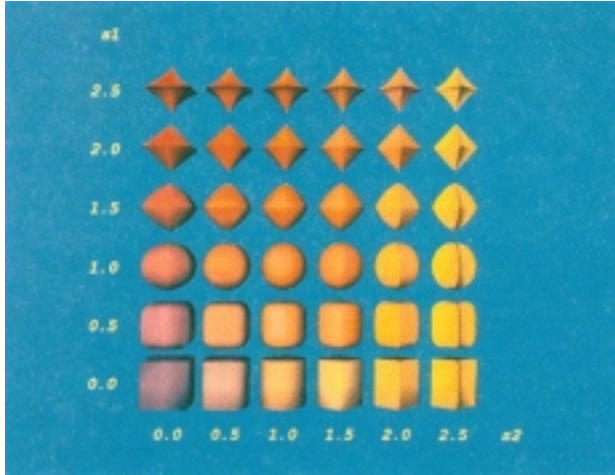



그림 10-13. 파라메터 s_1과 s_2의 서로 다른 값과 r_x=r_y=r_z에 의하여 표시되는 초타원체

5절. 무정형물체

어떤 물체들은 고정된 형태를 유지하지 않으면서 일정한 운동 또는 다른 물체에 접근할 때 자기 겉면의 특성을 변화시킨다. 이런 부류의 물체들의 실례로는 분자구조, 물방울 기타 흐름현상, 녹는 물체, 인체의 근육 등을 들수 있다. 이런 물체들은 《무정형성》(blobbiness)으로 표현할수 있으며 그것들의 형태는 어느 정도 류동성을 보여 주기때문에 간단히 **무정형물체**(blobby object)라고 한다.

실례로 분자의 형태는 고립적일 때에는 구로 표현할수 있지만 분자가 다른 분자에 다가갈 때에는 그의 형태가 변한다. 전자구름밀도의 형태가 변하는것은 두 분자사이에 일어 나는 《속박》때문이다. 그림 10-14에서는 두개의 분자가 멀어 져 갈 때 분자들의 형태에서의 잡아당기기효과, 툭 끊어지기효과, 수축효과를 보여 주었다. 이런 특성들은 단순히 구 또는 타원을 가지고서는 적절하게 표현할수 없다. 그림 10-15에서 보여 준 사람의 근육형태도 류사한 특성을 보여 준다. 이 경우에는 면의 형태를 전체 체적이 일정하게 되도록 모형화할수 있다.

그림 10-14. 분자의 결합(두 분자가 서로 멀어 져 갈 때 면의 형태는 잡아 당겨 지다가 툭 끊어 지며 마지막에 구로 수축된다.)

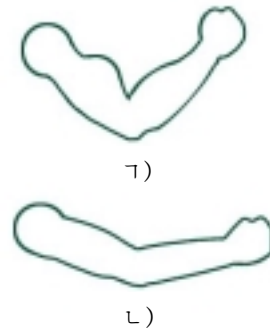



그림 10-15. 사람팔의 무정형적인 근육형태

공간령역에서의 분포함수와 같이 무정형물체를 표현하는 여러가지 방법들이 개발되였다. 이렇게 하는 한가지 방법은 물체를 가우스밀도함수(정규밀도함수) 즉 《봉우리》(그림 10-16)들의 결합으로 모형화하는것이다. 이때 면함수는

$$f(x,y,z)=\sum_k b_k e^{-a_k r_k^2}-T=0 \qquad (10-17)$$

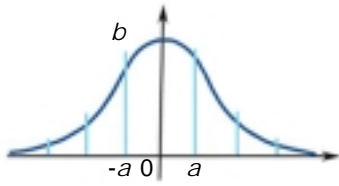



그림 10-16. 중심이 위치 0에 있고 높이가 b, 표준편차가 a 인 3차원가우스봉우리

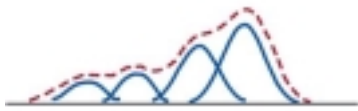



그림 10-17. 4개의 가우스봉우리로 형성되는 합성무정형물체

와 같이 정의된다. 여기서 $r=\sqrt{x^2+y^2+z^2}$, 파라메터 T는 지적된 턱값, 파라메터 a와 b는 개별적인 물체의 무정형량을 조정하는데 리용된다. 파라메터 b에 대한 부의 값은 봉우리대신에 골짜기를 만드는데 리용된다. 그림 10–17에서는 4개의 가우스밀도함수로 모형화한 합성물체의 면의 구조를 보여 주었다. 턱값준위에서 수값뿌리를 구하는 수법은 사귐점들의 자리표값을 찾아 내는데 리용된다. 개별적인 물체의 자름면은 원 또는

타원으로 모형화한다. 두개의 자름면이 서로 가까워 지면 그림 10-14에서와 같이 그것들은 병합되여 하나의 무정형형태를 형성하는데 형태의 구조는 두 물체사이의 거리에 관계된다.

무정형물체를 만드는 다른 방법은 정해 진 구간밖에서는 지수함수적으로 감소하지 않고 0으로 되는 밀도함수를 리용한다. 《메타볼》모형은 합성물체를

$$f(r)=\begin{cases} b(1-3r^2/d^2), & 0<r\le d/3 \\ \frac{3}{2}b(1-r/d)^2, & d/3<r\le d \\ 0, & r>d \end{cases} \qquad (10-18)$$

형식의 2차밀도함수의 결합으로 표현한다. 그리고 《유연한 물체》모형은 함수

$$f(r)=\begin{cases} 1-\frac{22r^2}{9d^2}+\frac{17r^4}{9d^4}-\frac{4r^6}{9d^6}, & 0<r\le d \\ 0, & r>d \end{cases} \qquad (10-19)$$

를 리용한다.

일부 설계 및 그림그리기프로그람들에서는 다각형 또는 스플라인함수만으로는 적절하게 모형화할수 없는 응용처리를 위하여 무정형함수모형화를 제공하고 있다. 그림 10-18에서는 메타볼을 리용하는 무정형물체모형작성기의 사용자대면부를 보여 주었다.

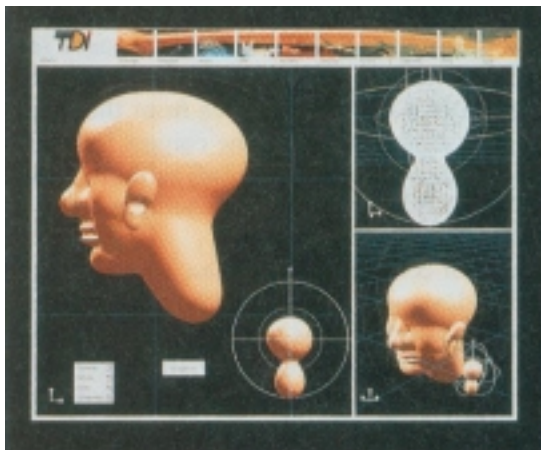



그림 10-18. 메타볼에 의하여 물체를 모형화할 때 Blob Modeler 및 Blob Animator 프로그람에서 사용되는 화면배치

6절. 스플라인표현

제도에서 스플라인은 지적된 점들의 모임을 지나는 미끈한 곡선을 만드는데 리용하는 유연한 좁고 긴 띠모양의 자이다. 곡선을 그릴 때에는 제도판에 이 자를 고정시키기 위하여 여러개의 작은 추들을 이 자의 길이에 따라 올려 놓는다. 용어 스플라인곡선은 본래 이 방법에 의하여 그려 지는 곡선을 말하였다. 이런 곡선은 이 곡선의 여러 구간을 지나 가면서 1차 및 2차도함수가 련속인 부분구간3차다항식함수에 의하여 수학적으로 표현할수 있다. 지금 콤퓨터도형처리에서의 용어 **스플라인곡선**(spline curve)은 부분곡선들의 경계에서 지적된 련속성조건들을 만족시키는 다항식들에 의하여 형성되는 복합곡선을 가리킨다. **스플라인곡면**(spline surface)은 직교하는 두 스플라인곡선들의 모임으로 표현할수 있다. 도형처리응용에서는 여러가지 방법으로 스플라인을 지적한다. 매개 지적에서는 일정한 경계조건들과 함께 개별적인 다항식의 형태를 간단히 가리킨다.

스플라인은 도형처리응용에서 곡선과 곡면의 형태설계, 그림을 콤퓨터에 기억하기 위한 수자변환, 장면안의 물체 또는 카메라에 대한 동화경로지적에 리용된다. 스플라인의 대표적인 CAD응용에는 자동차의 본체설계, 비행기와 우주비행선의 겉면설계, 배의 선체설계들이 포함된다.

보간 및 근사스플라인

스플라인곡선은 **조종점**(control point)이라고 하는 곡선의 일반적인 형태를 가리키는 자리표위치들의 모임을 주는 방법으로 지적한다. 그다음 이 조종점들을 부분구간련속인 보조변수다항식들로 맞춘다. 한가지 방법은 그림 10-19에서와 같이 곡선이 매개 조종점을 통과하도록 다항식들을 맞추는 방법인데 이때 곡선은 조종점들의 모임을 보간한다고 말한다. 다른 방법은 곡선이 모든 조종점을 꼭 통과해야 하는것은 아니고 대체적으로 조종점경로를 따르도록 다항식들을 맞추는 방법인데 이때 곡선은 조종점들의 모임을 근사화한다고 말한다(그림 10-20).

그림 10-19. 부분구간련속다항식들에 의하여 보간되는 6개 조종점들의 모임

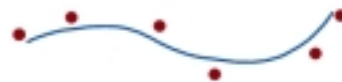



그림 10-20. 부분구간련속다항식들에 의하여 근사되는 6개 조종점들의 모임

보간곡선은 일반적으로 그림을 수자변환하거나 또는 동화경로를 지적하는데 리용된다. 근사곡선은 주로 물체의 겉면을 구조화하는 설계도구로 리용된다. 그림 10-21에서는 설계응용을 위하여 만든 근사스플라인곡면을 보여 주었다. 직선들은 곡면우에서 조종점위치들을 련결한다.

스플라인곡선은 조종점을 조작하여 정의하고 수정하며 처리한다. 설계자는 조종점들의 공간적인 위치를 대화식으로 선택하여 대체적인 곡선을 만들수 있다. 주어 진 조종점들의 모임에 대하여 다항식맞추기를 진행한후 설계자는 곡선의 형태를 수정하기 위하여 조종점들의 일부 또는 전체의 위치를

다시 정할수 있다. 또한 조종점들에 기하학적변환을 적용하여 곡선을 평행이동, 회전, 비례변환할수 있다. CAD프로그람들은 설계자가 곡선형태를 쉽게 조정할수 있도록 보충적인 조종점들을 삽입할수 있게 한다.

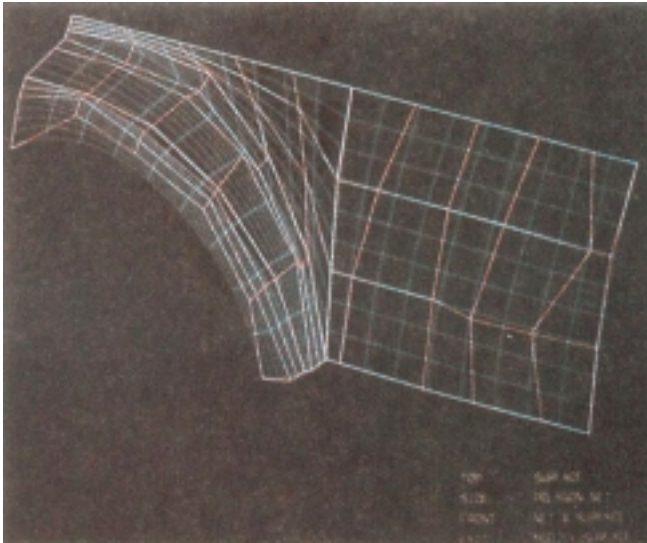



그림 10-21. 자동차설계를 위한 CAD 응용에서의 근사스플라인곡면(걸면의 륜곽은 다항식곡선들에 의하여 표시되며 걸면의 조종점들은 선분들로 련결되였다.)

조종점위치들을 둘러 싸는 볼록다각형의 테두리를 볼록페포(convex hull)라고 한다. 볼록페포의 형태를 상상하는 한가지 방법은 매개 조종점이 다각형의 둘레나 그의 내부에 들어 가도록 조종점들의 위치주위로 잡아 당긴 고무줄을 생각하는것이다(그림 10-22). 볼록페포는 조종점들을 련결하는 구역으로부터 곡선 또는 곡면이 리탈하는 한계를 준다. 볼록페포는 일부 스플라인들의 경계를 이루며 따라서 다항식들이 산만한 진동없이 조종점들을 원활하게 따른다는것을 보증한다. 또한 볼록페포의 다각형은 일부 알고리듬들에서 자르기구역으로 쓸수 있다.

근사스플라인에서는 보통 조종점들의 순서렬을 련결하는 절선을 현시하여 설계자가 조종점들의 순서를 알수 있게 한다. 이 련결된 선분들의 모임을 흔히 곡선의 조종그라프(control graph)라고 한다. 지적된 순서로 조종점들을 련결하는 선분들의 렬을 조종다각형(control polygon) 혹은 특성다각형(characteristic polygon)이라고 한다. 그림 10-23에서는 그림 10-22의 조종점렬에 대한 조종그라프의 형태를 보여 주었다.

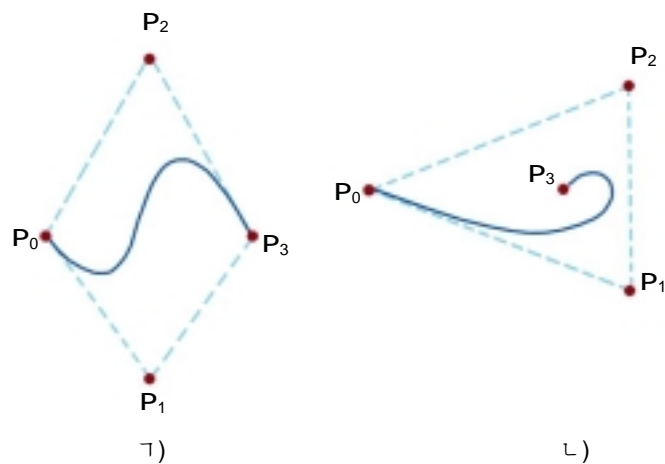



그림 10-22. 두개의 조종점들의 모임에 대한 볼록페포의 형태(파선)

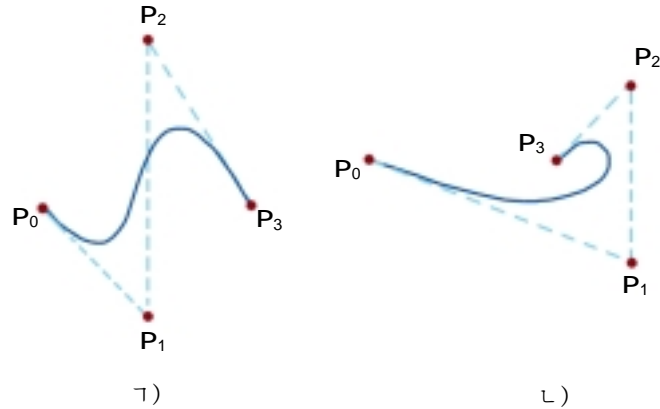



그림 10-23. 두 조종점들의 모임에 대한 조종그라프의 형태(파선)

보조변수련속성조건

보조변수곡선들을 원활하게 련결하기 위하여서는 련결점들에 여러가지 련속성조건들을 줄수 있다. 스플라인의 매개 부분을

$$x = x(u), \qquad y = y(u), \qquad z = z(u), \qquad u_1 \le u \le u_2 \qquad (10-20)$$

형식의 보조변수자리표함수들의 모임으로 표현하면 경계에서 린접하는 두 곡선토막들의 보조변수도함수를 정합시키는 방법에 의해 보조변수련속성(parametric continuity)을 설정할수 있다.

C^0련속성이라고 하는 0계보조변수련속성은 간단히 부분곡선들이 련결된다는것을 의미한다. 즉 첫번째 부분곡선의 u_2에서 평가되는 x, y, z의 값은 각각 다음부분곡선의 u_1에서 평가되는 x, y, z의 값과 같다는것이다. C^1련속성이라고 하는 1계보조변수련속성은 련결점에서 두개의 련속된 부분곡선에 대한 식 10-20의 자리표함수의 1계보조변수도함수(접선)들이 같다는것을 의미한다. 2계보조변수련속성 즉 C^2련속성은 련결점에서 두개의 부분곡선에 대한 1계 및 2계보조변수도함수들이 같다는것을 의미한다. 더 높은 계수의 보조변수련속성도 류사하게 정의된다. 그림 10-24에서는 C^0, C^1, C^2련속성의 실례를 보여 주었다.

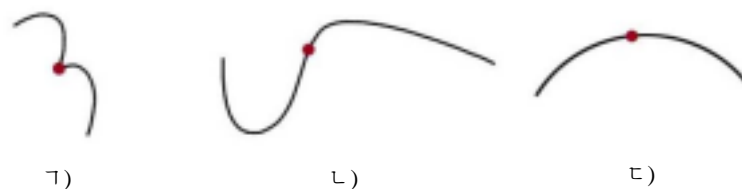



그림 10-24. 서로 다른 계수의 련속성을 리용하여
두개의 곡선토막을 련결한 곡선의 구성
ㄱ-0계련속성, ㄴ-1계련속성, ㄷ-2계련속성

2계련속성을 리용하여 련결되는 부분곡선들의 접선벡토르들의 변화속도는 련결점에서 같다. 그러므로 접선은 한 부분곡선에서부터 다른 부분곡선까지 원활하게 변한다(그림 10-24 ㄷ). 그러나 1계련속성을 리용할 때에는 두 부분곡선에 대한 접선벡토르들의 변화속도가 전혀 다를수 있다(그림 10-24 ㄴ). 따라서 린접한 두 부분곡선들의 일반적인 형태는 급격히 달라 질수 있다. 그림의 수자화나 일부 설계응용들에서는 1계련속성이면 충분하다. 그러나 카메라의 동화경로의 설정이나 많은 정밀CAD들에서는 2계련속성을 써야 한다. 파라메터 u를 등간격으로 변화시키면서 그림 10-24 ㄴ의 곡

선경로를 따라 카메라를 움직일 때 두 부분곡선의 경계에서 가속도가 급격히 변하여 운동렬에서 불련속성이 생기게 된다. 그러나 카메라를 그림 10-24 ㄷ의 경로를 따라 움직이면 운동의 프레임렬은 경계에서도 원할하게 변하게 된다.

기하학적련속성조건

련속된 두 부분곡선들을 련결하는 또 다른 방법은 **기하학적련속성조건**(geometric continuity)을 지적하는것이다. 이때에는 두 부분곡선들의 보조변수도함수들이 공통경계에서 서로 같지 않더라도 서로 비례되여야 한다.

G^0련속성이라고 하는 0계기하학적련속성은 0계보조변수련속성과 같다. 즉 경계점에서 두 부분곡선들이 같은 자리표위치를 가져야 한다. 1계기하학적련속성 즉 G^1련속성은 보조변수의 1계도함수들이 두 부분곡선들의 련결점에서 비례된다는것을 의미한다. 곡선의 보조변수위치를 $\mathbf{P}(u)$로 표시하면 G^1련속성하에서는 련결점에서 두 련속된 부분곡선들에 대한 접선벡토르 $\mathbf{P}'(u)$의 방향은 같지만 크기는 꼭 같지 않다. 2계기하학적련속성 즉 G^2련속성은 경계에서 두 부분곡선들의 1계 및 2계도함수가 다같이 비례된다는것을 의미한다. G^2련속성하에서 두개의 부분곡선들의 곡률은 련결위치에서 정합되게 된다.

기하학적련속성조건을 가지고 만든 곡선은 보조변수련속성조건을 가지고 만든것과 류사하지만 곡선의 형태에서 약간한 차이가 있다. 그림 10-25에서는 기하학적련속성과 보조변수련속성을 비교하였다. 기하학적련속성을 가진 곡선은 접선벡토르가 더 큰 부분쪽으로 당겨 진다.

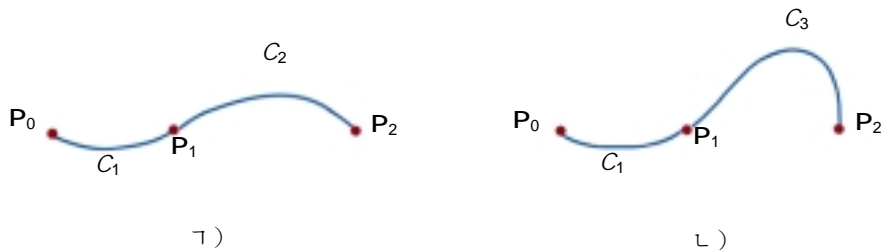



그림 10-25. ㄱ-보조변수련속성, ㄴ-기하학적련속성에 의하여 련결되는 두개의 부분곡선에 의하여 맞추어 지는 세개의 조종점(점 $\mathbf{p}_1$에서 곡선 C_3의 접선벡토르가 곡선 C_1의 접선벡토르보다 더 크다.)

스플라인지적

개별적인 스플라인표현은 다음의 세가지 등가적인 방법으로 지적할수 있다. 첫째로, 스플라인에 대한 경계조건들의 모임으로 줄수 있다. 둘째로, 스플라인의 특성행렬로 줄수 있다. 셋째로, 곡선경로에 따르는 위치들을 계산하는데서 지적된 기하학적제한들이 어떻게 결합되는가를 결정하는 혼합함수(blending function)(또는 토대함수(basis function))들의 모임으로 줄수 있다.

이 세가지 등가적인 지적들을 설명하기 위하여 스플라인이 부분구간들에서 x자리표에 대하여 다음의 보조변수3차다항식으로 표현된다고 하자.

$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x, \quad 0 \le u \le 1 \qquad (10-21)$$

이 곡선에 대한 경계조건들은 끝점들의 자리표 $x(0)$과 $x(1)$, 끝점들에서의 보조변수의 1계도함수 $x'(0)$과 $x'(1)$로 설정할수 있다. 이 4개의 경계조건들이면 4개의 결수 a_x, b_x, c_x, d_x의 값들을 결정하는데 충분하다.

경계조건들로부터 이 스플라인곡선의 특성행렬을 얻기 위하여 먼저 식 10-21을 행렬의 적

$$x(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} a_x \\ b_x \\ c_x \\ d_x \end{bmatrix} \tag{10-22}$$
$$= \mathbf{U} \cdot \mathbf{C}$$

으로 다시 쓴다. 여기서 **U**는 파라메터 u의 제곱들의 행벡토르이고 **C**는 결수들의 렬벡토르이다. 식 10-22를 리용하면 경계조건들을 행렬형식으로 쓰고 결수행렬 **C**에 대하여

$$\mathbf{C} = \mathbf{M}_{\text{spline}} \cdot \mathbf{M}_{\text{geom}} \tag{10-23}$$

와 같이 풀수 있다. 여기서 $\mathbf{M}_{\text{geom}}$은 스플라인에서의 기하학적인 제한값(경계조건)들을 포함하는 4개 요소의 렬벡토르이며 $\mathbf{M}_{\text{spline}}$은 기하학적인 제한값들을 다항식의 결수들로 변환하는 스플라인곡선의 4×4특성행렬이다. 행렬 $\mathbf{M}_{\text{geom}}$에는 조종점들의 자리표값과 기타 지적된 기하학적인 제한값들이 들어간다. 그러므로 식 10-22에서 **C**에 대한 행렬표현을 바꾸어 넣으면

$$x(u) = \mathbf{U} \cdot \mathbf{M}_{\text{spline}} \cdot \mathbf{M}_{\text{geom}} \tag{10-24}$$

를 얻을수 있다. 행렬 $\mathbf{M}_{\text{spline}}$은 때때로 토대행렬(basis matrix)이라고 하는데 스플라인표현을 특징지으며 한 스플라인표현을 다른 표현으로 변환하는데서 특히 쓸모 있다.

마지막으로 기하학적인 제한파라메터들로부터 자리표 x에 대한 다항식표현을 얻기 위하여 식 10-24를 전개하면

$$x(u) = \sum_{k=0}^{3} g_k \cdot BF_k(u) \tag{10-25}$$

여기서 g_k는 조종점의 자리표와 조종점에서의 곡선의 경사도와 같은 제한파라메터이며 $BF_k(u)$는 다항식으로 된 혼합함수이다. 다음 절들에서는 일반적으로 리용되는 몇가지 스플라인들과 그것들의 행렬 및 혼합함수를 설명한다.

7절. 3차스플라인보간방법

스플라인보간은 물체의 운동경로를 설정하거나 또는 현존하는 물체나 그림을 표현하는데 제일 많이 리용되며 때때로 물체의 형태를 설계하는데도 리용된다. 3차다항식은 적응성과 계산속도를 합리적으로 결합하고 있다. 더 높은 차수의 다항식에 비하여 3차스플라인은 계산량이 적고 기억기를 적게 요구하며 안정하다. 더 낮은 차수의 다항식에 대비하면 3차스플라인은 모든 형태의 곡선을 모형화할수 있다.

조종점들의 모임이 주어 지면 3차보간스플라인은 부분구간3차다항식곡선들이 매개 조종점을 통과하도록 곡선을 맞추는 방법으로 얻는다. 자리표

$$\mathbf{p}_k = (x_k, y_k, z_k), \qquad k = 0, 1, 2, \cdots, n$$

으로 지적되는 n+1개의 조종점이 있다고 하자. 이 점들에 대한 3차보간맞추기를 그림 10-26에서 보여 주었다. 조종점들의 매 쌍들사이를 맞추는 보조변수3차다항식은 다음의 련립방정식으로 표현할수 있다.

$$\begin{aligned} x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\ y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y, \qquad (0 \le u \le 1) \\ z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z \end{aligned} \tag{10-26}$$

n+1개 조종점들사이의 n개의 매 부분곡선들에 대한 이 세개 방정식의 4개의 결수 a, b, c, d의 값을 결정하여야 한다. 이를 위하여 모든 경우에 수값을 얻을수 있도록 부분곡선들사이의 련결에 대한 충분한 경계조건들을 설정한다. 다음 소제목에서 3차보간스플라인에 대하여 경계조건들을 설정하는 일반적인 방법들을 설명한다.

자연3차스플라인

도형처리응용을 위하여 개발된 첫 스플라인곡선들중의 하나는 자연3차스플라인(natural cubic spline)이다. 이 보간곡선은 본래의 제도스플라인의 수학적인 표현이다. 자연3차스플라인은 린접한 두개의 부분곡선이 공통경계에서 동일한 1계 및 2계보조변수도함수를 가지는 곡선이라고 정식화한다. 그러므로 자연3차스플라인은 C^2련속성을 가진다.

그림 10-26에서와 같이 만일 맞추어야 할 조종점들의 개수가 n+1개이라면 부분곡선들의 개수는 n개이며 결정하여야 할 다항식결수들의 총 개수는 $4n$개이다. n−1개의 각 내부조종점들에서 조종점의 량쪽에 있는 두개의 부분곡선은 그 조종점에서 동일한 1계 및 2계보조변수도함수를 가져야 하며 매개 곡선은 조종점을 통과해야 한다는 4개의 경계조건을 가진다. 이것은 $4n$개의 다항식결수들이 만족시켜야 할 $4n-4$개의 방정식을 준다. 첫번째 조종점 $\mathbf{p}_0$ 즉 곡선의 시작위치에서 하나의 추가적인 조건을 얻으며 곡선의 마지막점인 조종점 $\mathbf{p}_n$에서 다른 조건을 얻는다. 따라서 모든 결수들에 대한 값을 결정하기 위하여서는 아직 2개의 조건이 더 필요하다. 두개의 보충적인 조건을 얻는 한가지 방법은 $\mathbf{p}_0$과 $\mathbf{p}_n$에서 2계도함수를 0으로 설정하는것이다. 다른 방법은 본래의 조종점렬의 량끝에 각각 한개씩 두개의 《가짜》조종점들을 추가하는것이다. 즉 조종점 $\mathbf{p}_{-1}$과 조종점 $\mathbf{p}_{n+1}$을 추가한다. 그러면 본래의 모든 조종점들은 내부점으로 되며 필요한 $4n$개의 경계조건을 가진다.

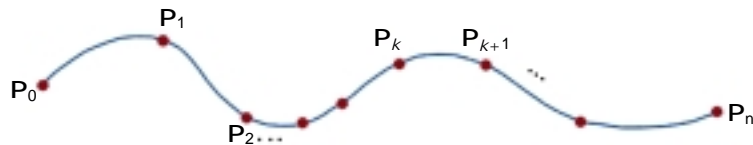



그림 10-26. n+1개 조종점들에 대한 부분구간련속3차스플라인보간

비록 자연3차스플라인은 제도스플라인의 수학적인 모형이지만 그것은 중요한 결함을 가진다. 임의의 하나의 조종점의 위치가 변경되면 전체 곡선이 영향을 받는다. 그러므로 자연3차스플라인은 국부적인 조종을 할수 없고 모든 조종점들을 새로 지적함이 없이는 곡선의 부분을 다시 구성할수 없다.

에르미트보간

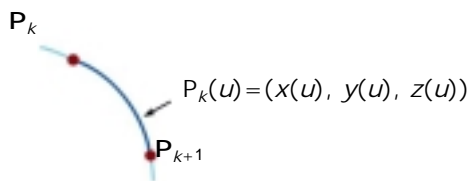



그림 10-27. 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$사이의 에르미트 부분곡선에 대한 보조변수점함수 $\mathbf{P}(u)$

프랑스 수학자 Charles Hermite의 이름을 붙인 에르미트스플라인(Hermite spline)은 매개 조종점에서 지적되는 접선에 의하여 보간하는 부분구간3차다항식이다. 자연3차스플라인과 달리 에르미트스플라인은 매개 부분곡선이 그의 끌점들에서의 제한조건에만 관계되기때문에 국부적으로 조정할수 있다.

그림 10-27에 보여 준바와 같이 만약 $\mathbf{P}(u)$가 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$사이의 부분곡선에 대한 보조변수점의 3차

함수라면 이 에르미트부분곡선을 정의하는 경계조건은

$$\begin{aligned} \mathbf{P}(0) &= \mathbf{p}_k \\ \mathbf{P}(1) &= \mathbf{p}_{k+1} \\ \mathbf{P}'(0) &= \mathbf{Dp}_k \\ \mathbf{P}'(1) &= \mathbf{Dp}_{k+1} \end{aligned} \tag{10-27}$$

이다. 여기서 $\mathbf{Dp}_k$와 $\mathbf{Dp}_{k+1}$은 각각 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$에서의 보조변수도함수(곡선의 경사도)값이다.

이 에르미트부분곡선에 대하여서는 식 10-26과 등가인 벡토르를

$$\mathbf{P}(u) = \mathbf{a}u^3 + \mathbf{b}u^2 + \mathbf{c}u + \mathbf{d}, \qquad 0 \le u \le 1 \tag{10-28}$$

와 같이 쓸수 있다. 여기서 $\mathbf{P}$의 x성분은 $x(u)=a_xu^3+b_xu^2+c_xu+d_x$이다. y와 z성분에 대해서도 류사하다. 식 10-28과 등가인 행렬은

$$\mathbf{P}(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{10-29}$$

이며 점함수의 도함수는

$$\mathbf{P}'(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{10-30}$$

와 같이 표현될수 있다. 앞의 두 방정식들에서 파라메터 u에 대하여 끝점값 0과 1을 대입하면 에르미트경계조건 10-27은 행렬형식

$$\begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \tag{10-31}$$

으로 표현할수 있다. 다항식결수들에 대하여 이 방정식을 풀면

$$\begin{aligned} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} \\ &= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} \\ &= \mathbf{M}_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} \end{aligned} \tag{10-32}$$

이다. 여기서 에르미트행렬 $\mathbf{M}_H$ 는 경계조건행렬의 역행렬이다. 따라서 식 10-29는 경계조건에 의하여

$$\mathbf{P}(u) = \begin{bmatrix} u^3 & u^3 & u & 1 \end{bmatrix} \cdot \mathbf{M}_H \cdot \begin{bmatrix} \mathbf{p}_k \\ \mathbf{p}_{k+1} \\ \mathbf{Dp}_k \\ \mathbf{Dp}_{k+1} \end{bmatrix} \qquad (10-33)$$

로 쓸수 있다.

마지막으로 다항식형식을 얻기 위하여 식 10-33에서 행렬곱하기를 진행하고 경계조건들에 대한 결수들을 한데 모으면 에르미트혼합함수에 대한 식을 결정할수 있다.

$$\begin{aligned}\mathbf{P}(u) &= \mathbf{p}_k(2u^3 - 3u^2 + 1) + \mathbf{p}_{k+1}(-2u^3 + 3u^2) + \mathbf{Dp}_k(u^3 - 2u^2 + u) \\ &\quad + \mathbf{Dp}_{k+1}(u^3 - u^2) \qquad (10-34) \\ &= \mathbf{p}_k H_0(u) + \mathbf{p}_{k+1} H_1(u) + \mathbf{Dp}_k H_2(u) + \mathbf{Dp}_{k+1} H_3(u)\end{aligned}$$

다항식 $H_k(u)$, k=0, 1, 2, 3은 곡선의 매개 자리표위치를 얻는데서 경계조건값(끌점의 자리표와 경사도)들을 혼합하기때문에 혼합함수라고 한다. 그림 10–28에서는 네가지 에르미트혼합함수의 형태를 보여주었다.

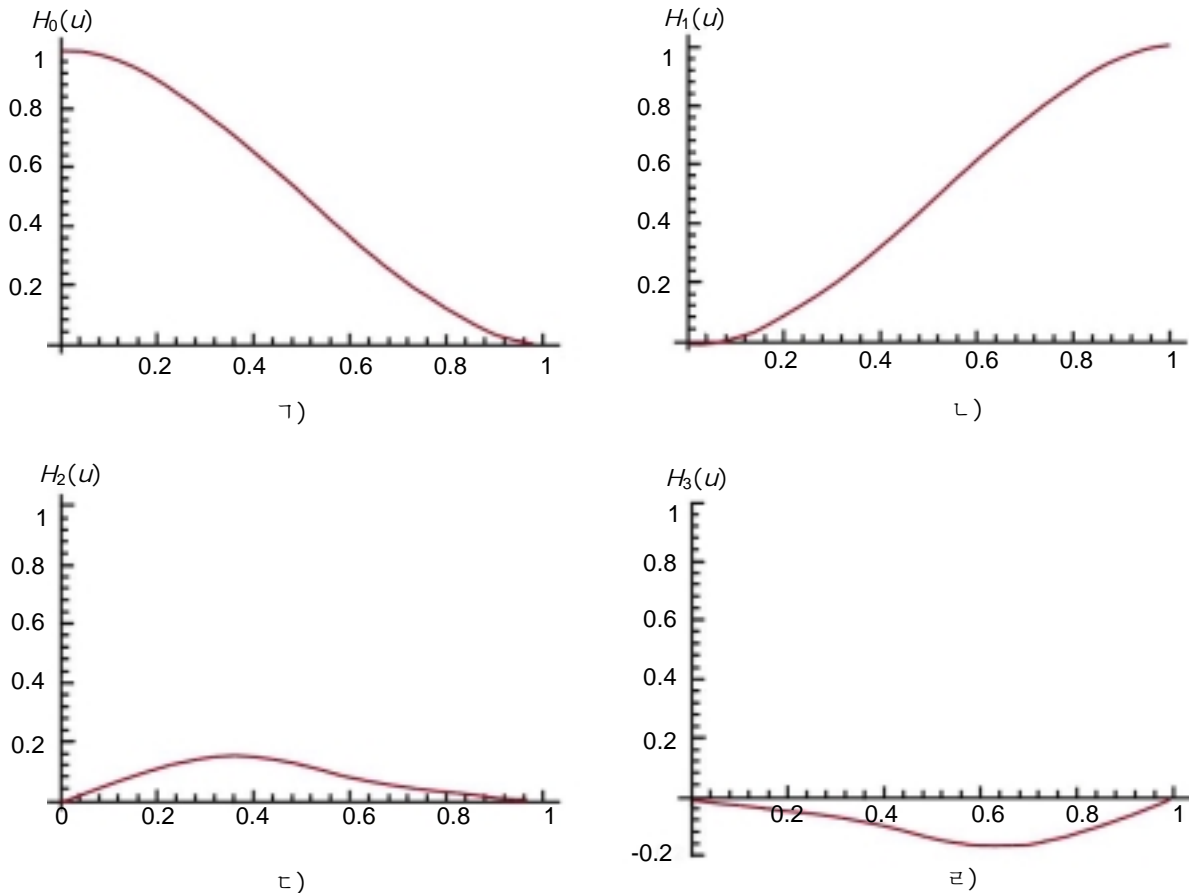



그림 10-28. 에르미트혼합함수

에르미트다항식은 곡선의 경사도를 지적하거나 근사화하는것이 그리 힘들지 않는 일부 수자변환응용들에서 쓸수 있다. 그러나 콤퓨터도형처리의 대부분의 문제들에서는 곡선의 경사도나 기타 기하학적인 정보에 대한 입력값들이 필요없이 조종점의 자리표만 가지고 스플라인곡선을 발생시키는것이

더 편리하다. 다음의 소제목에서 설명하는 기본스플라인과 코카니크-바텔스스플라인은 조종점들에서 곡선의 도함수값들을 요구하지 않는 에르미트스플라인의 변종이다. 이 스플라인들에서는 조종점들의 자리표위치로부터 보조변수도함수를 계산한다.

기본스플라인

에르미트스플라인과 마찬가지로 **기본스플라인**(cardinal spline)은 매 부분곡선의 끝점들에서 지적되는 접선들에 의하여 보간하는 부분구간3차함수이다. 차이점은 끝점들의 접선들에 대한 값을 주지 않아도 된다는데 있다. 기본스플라인에서는 조종점에서의 경사도값이 두개의 린접한 조종점의 자리표로부터 계산된다.

기본스플라인의 부분곡선은 련속된 4개의 조종점들에 의하여 완전히 지적된다. 중간의 두개 조종점은 부분곡선의 끝점들이며 다른 두개의 점은 끝점들의 경사도의 계산에 리용된다. 그림 10-29에서와 같이 $\mathbf{P}(u)$를 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$사이의 부분곡선에 대한 보조변수점의 3차함수라고 하면 $\mathbf{p}_{k-1}$부터 $\mathbf{p}_{k+2}$까지의 4개의 조종점은 기본스플라인에 대하여

$$\begin{aligned}
\mathbf{P}(0) &= \mathbf{p}_k \\
\mathbf{P}(1) &= \mathbf{p}_{k+1} \\
\mathbf{P}'(0) &= \frac{1}{2}(1-t)(\mathbf{p}_{k+1} - \mathbf{p}_{k-1}) \\
\mathbf{P}'(1) &= \frac{1}{2}(1-t)(\mathbf{p}_{k+2} - \mathbf{p}_k)
\end{aligned} \qquad (10-35)$$

와 같이 경계조건들을 설정하는데 리용된다. 그러므로 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$에서의 경사도는 각각 현 $\overline{\mathbf{p}_{k-1}\mathbf{p}_{k+1}}$와 $\overline{\mathbf{p}_k\mathbf{p}_{k+2}}$에 비례한다(그림 10-30). 파라메터 t는 기본스플라인이 입력조종점들을 얼마나 느슨하게 또는 팽팽하게 맞추는가를 조종하므로 **당김파라메터**(tension parameter)라고 한다. 그림 10-31에서는 당김파라메터 t의 대단히 작은 값과 대단히 큰 값에 대한 기본스플라인곡선의 형태를 보여주었다. t=0일 때의 기본스플라인을 카트물-롬(Catmull-Rom)스플라인 또는 오버하우써(Overhauser)스플라인이라고 한다.

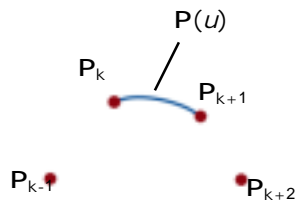



그림 10-29. 조종점 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$사이의 기본스플라인의 부분곡선에 대한 보조변수점의 함수 $\mathbf{P}(u)$

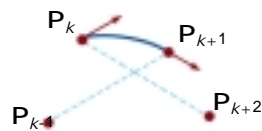



그림 10-30. 기본스플라인의 부분곡선의 끝점들에서의 접선벡토르는 린접한 조종점들에 의하여 형성되는 현(파선)에 비례한다.

그림 10-31. 기본스플라인의 부분곡선의 형태에 대한 당김파라메터의 효과

에르미트스플라인에서와 류사한 방법을 리용하여 경계조건에 대한 식 10-35를 행렬형식

$$\mathbf{P}(u)=\begin{bmatrix}u^3 & u^3 & u & 1\end{bmatrix}\cdot\mathbf{M}_C\cdot\begin{bmatrix}\mathbf{p}_{k-1}\\ \mathbf{p}_k\\ \mathbf{p}_{k+1}\\ \mathbf{p}_{k+2}\end{bmatrix} \tag{10-36}$$

으로 변환할수 있다. 여기서 기본행렬은

$$\mathbf{M}_C=\begin{bmatrix}-s & 2-s & s-2 & s\\ 2s & s-3 & 3-2s & -s\\ -s & 0 & s & 0\\ 0 & 1 & 0 & 0\end{bmatrix} \tag{10-37}$$

이며 $s=(1-t)/2$이다.

행렬방정식 10-36을 다항식형식으로 전개하면

$$\begin{aligned}\mathbf{P}(u)&=\mathbf{p}_{k-1}(-su^3+2su^2-su)+\mathbf{p}_k[(2-s)u^3+(s-3)u^2+1]\\ &\quad+\mathbf{p}_{k+1}[(s-2)u^3+(3-2s)u^2+su]+\mathbf{p}_{k+2}(su^3-su^2)\\ &=\mathbf{p}_{k-1}CAR_0(u)+\mathbf{p}_kCAR_1(u)+\mathbf{p}_{k+1}CAR_2(u)+\mathbf{p}_{k+2}CAR_3(u)\end{aligned} \tag{10-38}$$

이다. 여기서 다항식 $CAR_k(u)$, k=0, 1, 2, 3은 기본혼합함수이다. 그림 10–32에서는 t=0인 기본스플라인에 대한 기본혼합함수들의 모양을 보여 주었다.

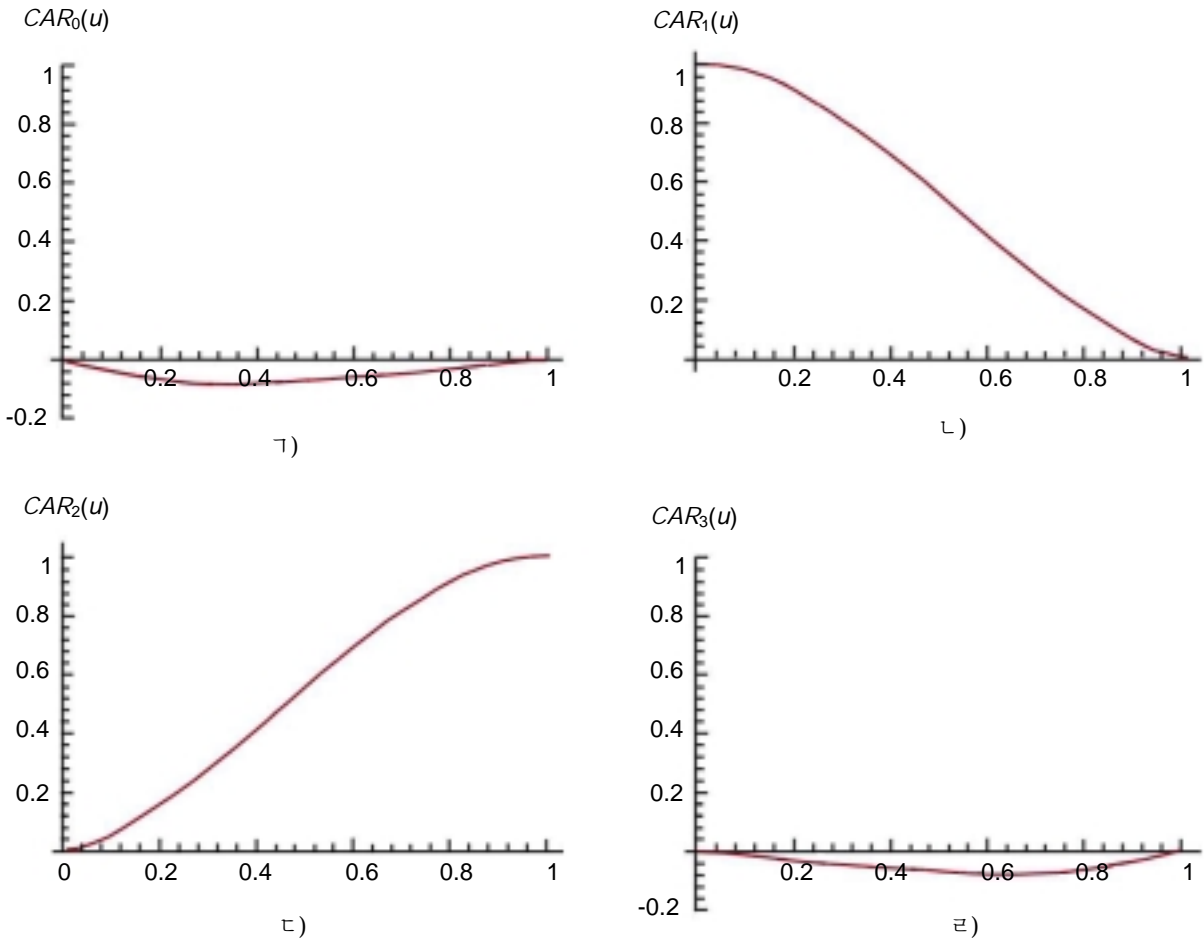



그림 10-32. t=0, s=0.5에 대한 기본혼합함수

코카니크-바텔스스플라인

이 보간3차다항식은 기본스플라인의 확장이다. 부분곡선들의 형태를 자유롭게 조정하기 위하여 코카니크-바텔스(Kochanek-Bartels)스플라인을 정의할 때 제한식에 두개의 파라메터를 보충적으로 도입한다.

주어 진 4개의 련속된 조종점을 각각 $\mathbf{p}_{k-1}$, $\mathbf{p}_k$, $\mathbf{p}_{k+1}$, $\mathbf{p}_{k+2}$로 표시하면 $\mathbf{p}_k$와 $\mathbf{p}_{k+1}$사이의 코카니크-바텔스부분곡선에 대한 경계조건은

$$
\begin{aligned}
\mathbf{P}(0) &= \mathbf{p}_k \\
\mathbf{P}(1) &= \mathbf{p}_{k+1} \\
\mathbf{P}'(0)_{\text{in}} &= \frac{1}{2}(1-t)[(1+b)(1-c)(\mathbf{p}_k - \mathbf{p}_{k-1}) \\
&\qquad + (1-b)(1+c)(\mathbf{p}_{k+1} - \mathbf{p}_k)] \\
\mathbf{P}'(1)_{\text{out}} &= \frac{1}{2}(1-t)[(1+b)(1+c)(\mathbf{p}_{k+1} - \mathbf{p}_k) \\
&\qquad + (1-b)(1-c)(\mathbf{p}_{k+2} - \mathbf{p}_{k+1})]
\end{aligned}
\tag{10-39}
$$

와 같이 정의한다. 여기서 t는 당김파라메터, b는 편위파라메터(bias parameter), c는 련속성파라메터(continuity parameter)이다. 코카니크-바텔스형식화에서는 보조변수도함수가 부분곡선의 경계에서 련속이 아닐수 있다.

당김파라메터 t는 기본스플라인에서와 같이 해석된다. 즉 그것은 부분곡선들의 느슨함 또는 팽팽함을 조종한다. 편위파라메터 b는 곡선이 부분곡선의 매 끝에서 구부러지는 량을 조정하는데 리용되며 따라서 부분곡선들을 어느 한 끝쪽으로 기울어 지게 할수 있다(그림 10-33). 파라메터 c는 부분곡선들의 경계에서 접선벡토르의 련속성을 조종한다. c에 령 아닌 값을 할당하면 부분곡선들의 경계에서 곡선의 경사도가 불련속으로 된다.

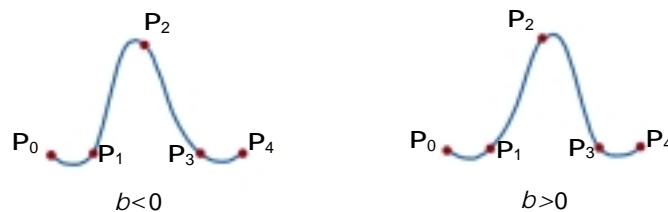



그림 10-33. 코카니크-바텔스스플라인의 부분곡선의 형태에 대한 편의파라메터의 효과

코카니크-바텔스스플라인은 동화의 경로를 모형화하기 위하여 설계되였다. 특히 물체의 운동에서의 갑작스러운 변화는 파라메터 c에 령 아닌 값을 주어 모의할수 있다.

8절. 베지에곡선과 베지에곡면

이 스플라인근사방법은 프랑스 기사 Pierre Bézier가 Renault자동차의 본체설계에 리용하기 위하여 개발하였다. 베지에(Bézier)스플라인은 곡선 및 곡면설계에 아주 쓸모 있고 편리한 많은 특성들을 가지며 실현하기도 쉽다. 이런 리유로 하여 베지에스플라인은 여러가지 CAD체계, 일반도형처리프로그람(씰리콘그라픽스체계의 GL과 같은), 작도 및 그림그리기프로그람(Aldus SuperPaint 및 Cricket Draw와 같은)들에서 널리 리용되고 있다.

베지에곡선

일반적으로 베지에부분곡선은 임의의 개수의 조종점들로 맞출수 있다. 근사될 조종점들의 개수와 그것들의 상대적인 위치는 베지에다항식의 차수를 결정한다. 보간스플라인과 마찬가지로 베지에곡선도 경계조건, 특성행렬, 혼합함수에 의하여 지적할수 있다. 일반베지에곡선은 혼합함수로 지적하는것이 제일 편리하다.

n+1개의 조종점위치 $\mathbf{p}_k=(x_k, y_k, z_k)$, $k=\overline{0,n}$ 가 주어 졌다고 하자. 이 자리표점들을 혼합하여 $\mathbf{p}_0$과 $\mathbf{p}_n$ 사이의 근사베지에다항식함수의 경로를 나타내는 다음의 위치벡토르 $\mathbf{P}(u)$를 만들수 있다.

$$\mathbf{P}(u)=\sum_{k=0}^{n}\mathbf{p}_k BEZ_{k,n}(u), \qquad 0\le u\le 1 \qquad (10-40)$$

베지에혼합함수 $BEZ_{k,n}(u)$는 베른슈타인다항식이다.

$$BEZ_{k,n}(u)=C(n,k)u^k(1-u)^{n-k} \qquad (10-41)$$

여기서 $C(n,k)$는 2항결수이다.

$$C(n,k)=\frac{n!}{k!(n-k)!} \qquad (10-42)$$

베지에혼합함수는 등가적으로

$$BEZ_{k,n}(u)=(1-u)BEZ_{k,n-1}(u)+uBEZ_{k-1,n-1}(u), \qquad n>k\ge 1 \qquad (10-43)$$

의 재귀계산으로 정의할수 있다. 여기서 $BEZ_{k,k}=u^k$, $BEZ_{0,k}=(1-u)^k$이다. 벡토르방정식 10-40은 개별적인 곡선자리표에 대한 세개의 보조변수방정식들의 모임을 표현한다.

$$x(u)=\sum_{k=0}^{n}x_k BEZ_{k,n}(u)$$
$$y(u)=\sum_{k=0}^{n}y_k BEZ_{k,n}(u) \qquad (10-44)$$
$$z(u)=\sum_{k=0}^{n}z_k BEZ_{k,n}(u)$$

일반적으로 베지에곡선은 사용하는 조종점의 개수보다 하나 작은 차수의 다항식이다. 즉 세개의 점은 포물선, 네개의 점은 3차곡선을 만든다. 그림 10-34에서는 xy평면(z=0)에서 여러가지로 선택된 조종점들에 대한 몇가지 베지에곡선의 모양을 보여 주었다. 그런데 조종점의 어떤 배치에서는 퇴화된 베지에다항식이 얻어 진다. 실례로 같은 직선우에 있는 세개의 조종점에 의하여 발생되는 베지에곡선은 선분이다. 그리고 모두가 동일한 자리표위치에 있는 조종점들의 모임은 하나의 점인 베지에《곡선》을 만든다.

베지에곡선은 실현하기가 쉽고 곡선설계에서 대단히 유력하기때문에 CAD체계뿐아니라 그림그리기 및 작도프로그람들에서 널리 리용되고 있다. 베지에곡선을 따라 자리표위치들을 효과적으로 결정하기 위하여 재귀계산을 리용한다. 실례로 련속된 2항결수들은 $n\ge k$에 대하여

$$C(n,k)=\frac{n-k+1}{k}C(n,k-1) \qquad (10-45)$$

와 같이 계산할수 있다. 다음의 실례프로그람은 베지에곡선을 발생시키는 방법을 보여 주었다.

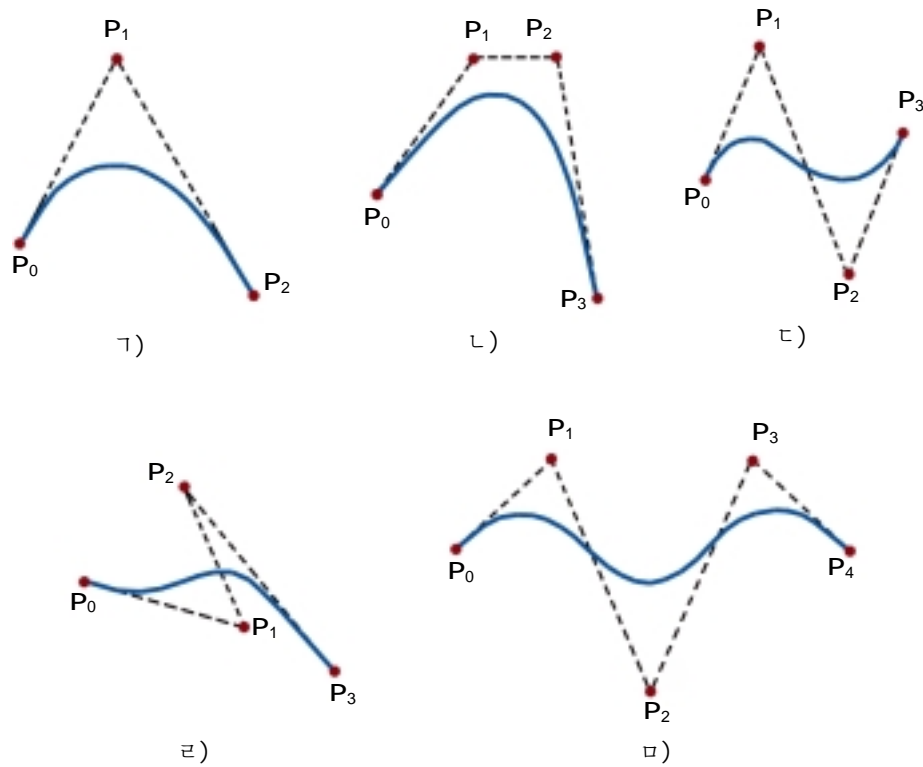



그림 10-34. 3, 4, 5개의 조종점들로 발생시킨 2차원베지에곡선의 실례들(파선은 조종점위치들을 련결한다.)

```
#include <math.h>
#include "graphics.h"

void computeCoefficients (int n, int * c)
{
  int k, i;

  for (k=0; k<=n; k++) {
    /* Compute n!/(k!(n-k)!) */
    c[k] = 1;
    for (i=n; i>=k+1; i--)
      c[k] *= i;
    for (i=n-k; i>=2; i--)
      c[k] /= i;
  }
}

void computePoint
  (float u, wcPt3 * pt, int nControls, wcPt3 * controls, int * c)
{
  int k, n = nControls - 1;
  float blend;
  pt->x = 0.0; pt->y = 0.0; pt->z = 0.0;
```

```
    /* Add in influence of each control point */
    for (k=0; k<nControls; k++) {
      blend = c[k] * powf (u, k) * powf (1-u, n-k);
      pt->x += controls[k]. x * blend;
      pt->y += controls[k]. y * blend;
      pt->z += controls[k]. z * blend;
    }
  }

  void bezier (wcPt3 * controls, int nControls, int m, wcPt3 * curve)
  {
    /* Allocate space for the coefficients */
    int * c = (int *) malloc (nControls * sizeof (int));
    int i;

    computeCoefficients (nControls-1, c);
    for (i=0; i<=m; i++)
      computePoint (i / (float) m, &curve[i], nControls, controls, c);
    free (c);
  }
```

베지에곡선의 특성

베지에곡선의 대단히 편리한 특성은 그것이 항상 첫번째와 마지막조종점을 통과한다는것이다. 즉 곡선의 두 끝에서 경계조건은

$$\mathbf{P}(0) = \mathbf{p}_0 \qquad \mathbf{P}(1) = \mathbf{p}_n \tag{10-46}$$

이다. 끝점들에서 베지에곡선의 보조변수1계도함수값은 조종점들의 자리표로부터

$$\mathbf{P}'(0) = -n\mathbf{p}_0 + n\mathbf{p}_1 \qquad \mathbf{P}'(1) = -n\mathbf{p}_{n-1} + n\mathbf{p}_n \tag{10-47}$$

와 같이 계산된다. 그러므로 곡선의 시작점에서 접선은 첫 두 조종점을 련결하는 선이며 곡선의 끝점에서 접선은 마지막 두 조종점을 련결하는 선이다. 류사하게 끝점들에서 베지에곡선의 보조변수2계도함수는

$$\mathbf{P}''(0) = n(n-1)[(\mathbf{p}_2 - \mathbf{p}_1) - (\mathbf{p}_1 - \mathbf{p}_0)] \qquad \mathbf{P}''(1) = n(n-1)[(\mathbf{p}_{n-2} - \mathbf{p}_{n-1}) - (\mathbf{p}_{n-1} - \mathbf{p}_n)] \tag{10-48}$$

와 같이 계산된다.

베지에곡선의 다른 중요한 특성은 그것이 조종점들의 볼록페포(볼록다각형경계)안에 있다는것이다. 이것은 베지에혼합함수들이 모두 정의 값을 가지며 그것들의 합은 항상 1이기때문이다.

$$\sum_{k=0}^{n} BEZ_{k,n}(u) = 1 \tag{10-49}$$

그러므로 곡선의 임의의 위치는 단순히 조종점위치들의 무게붙은 합으로 된다. 베지에곡선의 볼록페포특성은 다항식이 조종점들을 산만한 진동이 없이 원활하게 따른다는것을 보증하여 준다.

베지에곡선을 리용한 설계수법

닫긴 베지에곡선은 그림 10-35에 보여 준 실례에서와 같이 첫번째와 마지막조종점을 동일한 위치로 지적하여 만든다. 또한 한 자리표위치에 여러개의 조종점들을 지적하는 방법으로 그 위치에 더 많은 무게를 준다. 그림 10-36에서와 같이 동일한 자리표위치에 두개의 조종점을 입력하면 결과곡선은 이 위치에 더 가깝게 끌리운다.

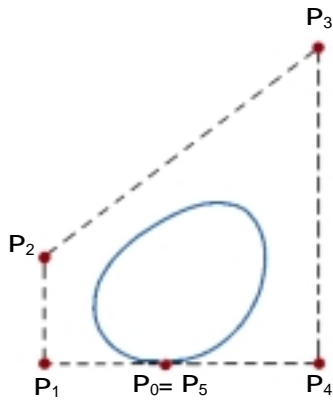



그림 10-35. 첫번째와 마지막조종점을 같은 위치에 지적하여 만든 닫긴 베지에곡선

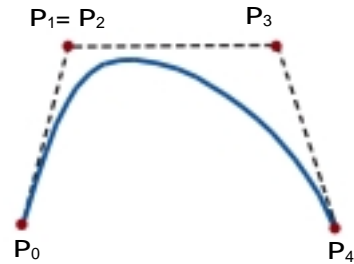



그림 10-36. 한 자리표위치에서 여러개의 조종점들을 지적하면 베지에곡선은 그 위치에 더 가깝게 끌리운다.

베지에곡선은 임의의 개수의 조종점들에 맞출수 있지만 이것은 더 높은 차수의 다항식함수계산을 요구한다. 복잡한 곡선은 보다 낮은 차수의 여러개의 베지에부분곡선들을 련결하여 만들수 있다. 곡선을 더 작은 부분곡선들로 이어 붙이면 작은 구간에서 곡선의 형태를 더 잘 조종할수 있다. 베지에곡선은 끝점들을 통과하기때문에 부분곡선들을 정합시키는것이(0계련속성) 쉽다. 또한 베지에곡선은 끝점에서의 접선이 그 끝점과 린접한 조종점을 련결하는 선이라는 중요한 특성을 가진다. 따라서 부분곡선들사이에 1계련속성을 얻으려면 새 구간의 조종점 $\mathbf{p}'_0$와 $\mathbf{p}'_1$을 앞구간의 조종점 $\mathbf{p}_{n-1}$ 및 $\mathbf{p}_n$과 같은 직선우에 있도록 선택하면 된다(그림 10-37). 두개의 부분곡선이 동일한 개수의 조종점을 가질 때에는 새 구간의 첫번째 조종점을 앞구간의 마지막조종점으로 선택하고 새 구간의 두번째 조종점의 위치는

$$\mathbf{p}_n + (\mathbf{p}_n - \mathbf{p}_{n-1})$$

에 정하면 C^1련속성이 얻어 진다. 이렇게 되면 세개의 조종점은 같은 직선우에 있으며 등간격을 가진다.

새 구간의 세번째 조종점의 위치를 앞구간의 마지막3개 조종점들의 위치로부터

$$\mathbf{p}_{n-2} + 4(\mathbf{p}_n - \mathbf{p}_{n-1})$$

와 같이 계산하면 두 베지에부분곡선들사이는 C^2련속성을 가진다. 베지에부분곡선들사이에 2계련속성을 요구하는것은 불필요한 제한으로 될수도 있다. 이것은 특히 매 구간에서 4개의 조종점만을 가지는 3차곡선에서 그렇게 된다. 이 경우 2계련속성은 첫 3개 조종점의 위치는 고정하고 곡선토막의 형태를 조종하는데 리용할수 있는 한개의 점만을 남긴다.

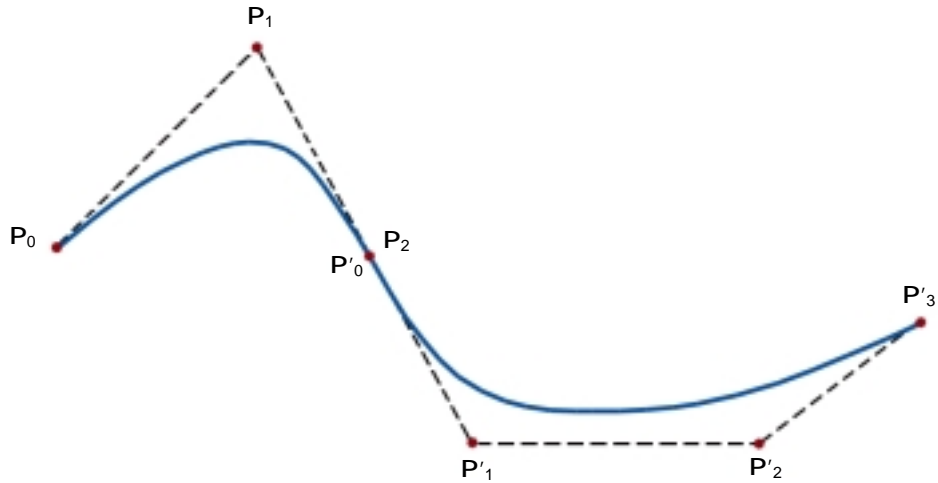



그림 10-37. 두개의 베지에부분곡선들로 형성된 부분구간근사곡선(부분곡선들사이에 $\mathbf{p}'_0 = \mathbf{p}_2$ 로 설정하고 점 $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}'_1$ 를 같은 직선우에 놓음으로써 0계 및 1계련속성을 얻는다.)

3 차베지에곡선

많은 도형처리프로그람들은 3차스플라인함수만을 제공한다. 이렇게 하면 설계를 합리적으로, 자유롭게 할수 있으면서도 더 높은 차수의 다항식을 쓸 때보다 계산을 줄인다. 3차베지에곡선은 4개의 조종점에 의하여 만든다. 식 10-41에 n=3을 대입하여 얻어 지는 3차베지에곡선에 대한 4개의 혼합함수는

$$
\begin{aligned}
BEZ_{0,3}(u) &= (1-u)^3 \\
BEZ_{1,3}(u) &= 3u(1-u)^2 \\
BEZ_{2,3}(u) &= 3u^2(1-u) \\
BEZ_{3,3}(u) &= u^3
\end{aligned}
\qquad (10-50)
$$

이다. 4개의 3차베지에혼합함수들의 곡선모양을 그림 10-38에 주었다. 혼합함수들의 모양은 조종점들이 파라메터 u가 0~1사이의 값을 가질 때 곡선의 형태에 어떻게 영향을 미치는가 하는것을 결정한다. u=0에서 령 아닌 혼합함수는 $BEZ_{0,3}$뿐이며 값 1을 가진다. u=1에서 령 아닌 혼합함수는 $BEZ_{3,3}$뿐이며 값 1을 가진다. 그러므로 3차베지에곡선은 항상 조종점 $\mathbf{p}_0$과 $\mathbf{p}_3$를 지나게 된다. 다른 함수 $BEZ_{1,3}$와 $BEZ_{2,3}$는 파라메터 u의 중간값들에서 곡선의 형태에 영향을 미치며 따라서 결과곡선은 점 $\mathbf{p}_1$과 $\mathbf{p}_2$쪽으로 향한다. 혼합함수 $BEZ_{1,3}$는 u=1/3에서 최대이며 $BEZ_{2,3}$는 u=2/3에서 최대이다.

그림 10-38에서 4개의 매 혼합함수가 파라메터 u의 전체 범위에서 령이 아니라는것을 주의하자. 따라서 베지에곡선은 곡선형태의 국부조종을 하지 못한다. 임의의 한 조종점의 위치를 다시 정하면 전체 곡선이 영향을 받게 된다.

3차베지에곡선의 끝위치에서 보조변수1계도함수(경사도)는

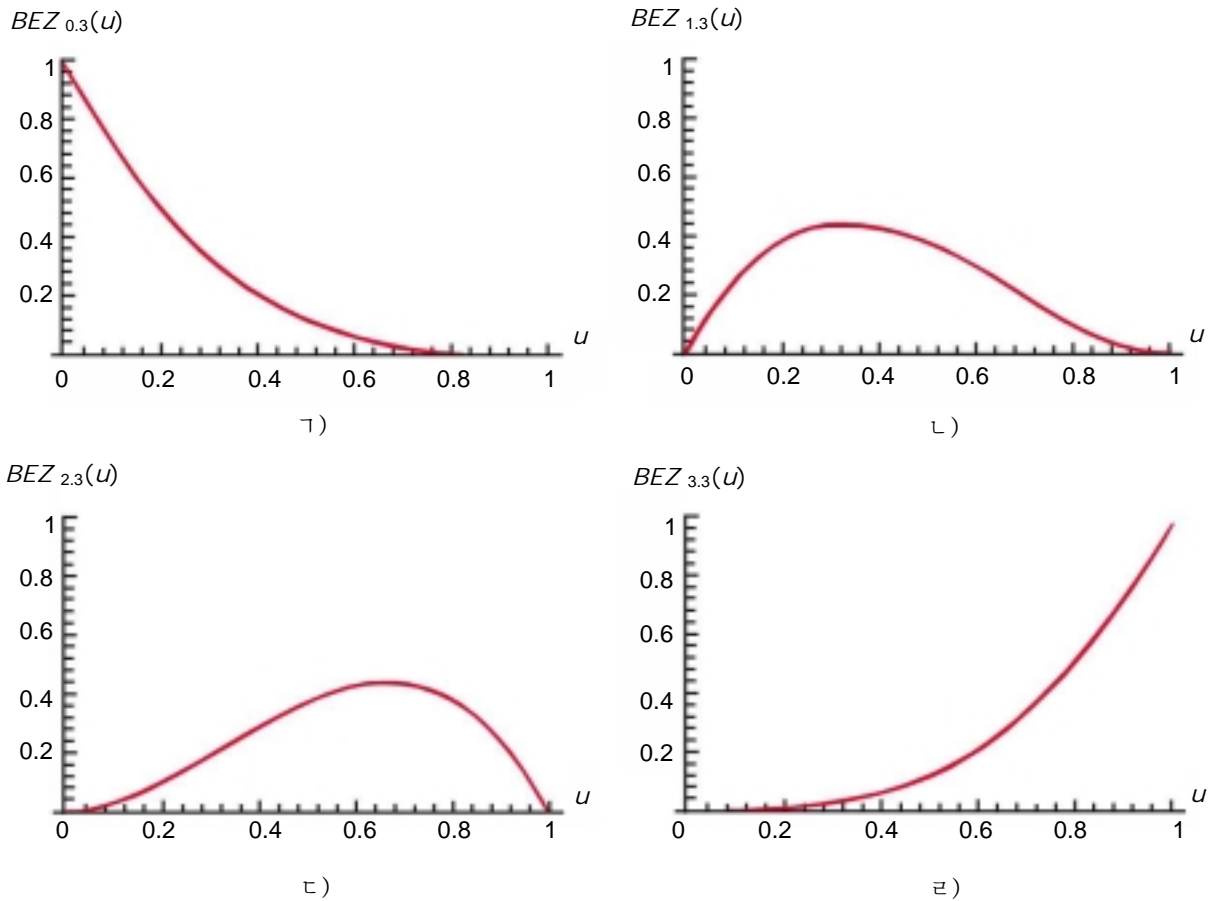



그림 10-38. 3차곡선(n=3)에 대한 4개의 베지에혼합함수

$$\mathbf{P}'(0) = 3(\mathbf{p}_1 - \mathbf{p}_0), \qquad \mathbf{P}'(1) = 3(\mathbf{p}_3 - \mathbf{p}_2)$$

이다. 그리고 보조변수2계도함수는

$$\mathbf{P}''(0) = 6(\mathbf{p}_0 - 2\mathbf{p}_1 + \mathbf{p}_2), \qquad \mathbf{P}''(1) = 6(\mathbf{p}_1 - 2\mathbf{p}_2 + \mathbf{p}_3)$$

이다. 보조변수도함수에 대한 이 식들은 부분곡선들사이에 C^1 또는 C^2련속성을 가지는 부분구간곡선을 만드는데 리용할수 있다.

혼합함수에 대한 다항식들을 전개하면 3차베지에점함수를 행렬형식

$$\mathbf{P}(u) = \begin{bmatrix} u^3 & u^3 & u & 1 \end{bmatrix} \cdot \mathbf{M}_{\mathrm{Bez}} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} \qquad (10-51)$$

으로 쓸수 있다. 여기서 **베지에행렬**은

$$\mathbf{M}_{\text{Bez}} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad (10-52)$$

이다. 보간스플라인에서와 같이 곡선의 《당김》과 《편위》를 조종하게 하는 보충적인 파라메터들도 도입할수 있다. 그러나 β스플라인은 물론 보다 더 편리한 B스플라인에서도 이 능력을 제공해 주고 있다.

베지에곡면

조종점들의 그물을 입력하여 직교하는 베지에곡선들의 두 모임을 지적할수 있다. 이것들은 물체면을 설계하는데 리용할수 있다. 베지에곡면에 대한 보조변수벡토르함수는 베지에혼합함수들의 적으로 만들어 진다.

$$\mathbf{P}(u,v) = \sum_{j=0}^{m}\sum_{k=0}^{n}\mathbf{p}_{j,k}BEZ_{j,m}(v)BEZ_{k,n}(u) \qquad (10-53)$$

여기서 $\mathbf{p}_{j,k}$는 $(m+1)\times(n+1)$개 조종점들의 위치를 가리킨다.

그림 10-39에서는 두개의 베지에곡면을 보여 주었다. 조종점들은 파선으로 련결시키고 실선은 u와 v가 상수인 곡선을 보여 준다. u가 상수인 매개 곡선은 v를 0~1사이 구간에서 변화시켜 표시하고 이때 u는 이 단위구간의 값들중 하나에 고정시킨다. v가 상수인 곡선도 마찬가지로 표시하였다.

베지에곡면은 베지에곡선과 같은 특성을 가지며 대화식설계응용에 편리한 방법을 제공한다. 걸면의 매개 곡면쪼각에 대하여 xy《바닥》평면에서 조종점들의 한개 그물을 선택하고 다음에 조종점들의 z자리표값에 대하여 바닥평면에서의 높이를 선택할수 있다. 곡면쪼각들은 다음에 경계조건을 리용하여 련결시킬수 있다.

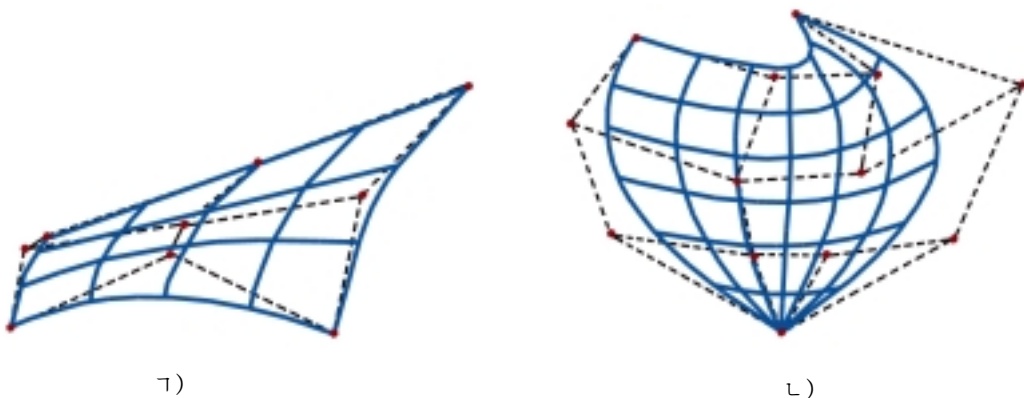



그림 10-39. m=3, n=3(ㄱ), m=4, n=4(ㄴ)일 때 만들어 지는 베지에곡면(파선은 조종점들의 련결이다.)

그림 10-40에서는 두개의 베지에부분곡면으로 형성되는 면을 보여 주었다. 곡선에서와 마찬가지로 두 부분곡면을 원활하게 련결하기 위하여 경계선에서 0계 및 1계련속성을 다같이 설정한다. 0계련속성은 경계에서 조종점들을 맞추어 얻는다. 1계련속성은 경계를 지나가는 직선상에서 조종점들을 선택하고 두 부분곡면의 경계를 지나는 지적된 조종점들의 매 모임에 대한 공직선상에서 토막들의 비를 일정하게 유지하여 얻는다.

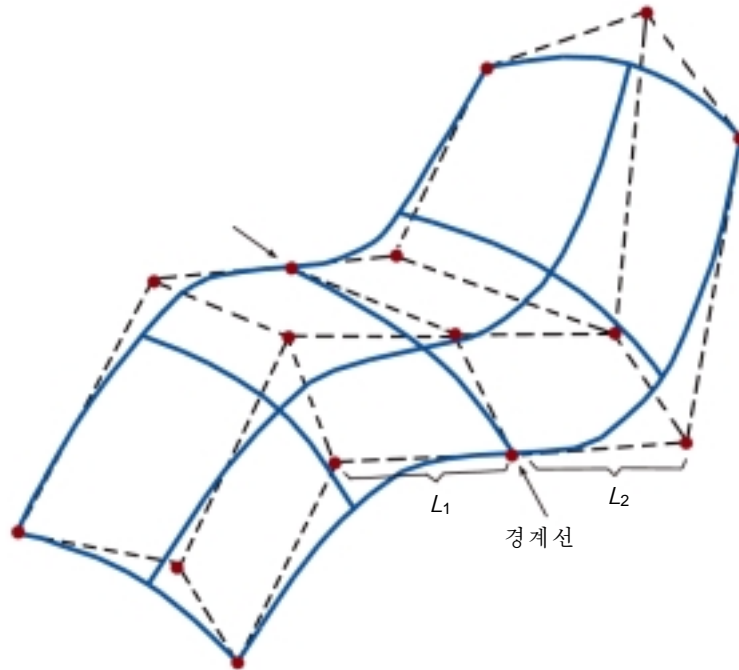



그림 10-40. 지적된 경계선에서 두개의 베지에부분곡면을 련결하여 만든 합성베지에곡면(파선은 지적된 조종점들을 련결시킨것이다. 1계련속성은 부분곡면들사이의 경계를 지나는 조종점들의 매 공직선에 대하여 길이 L_1과 길이 L_2의 비를 일정하게 하여 설정한다.)

9절. B스플라인곡선과 B스플라인곡면

B스플라인(B-spline)은 제일 널리 쓰이는 근사스플라인이다. B스플라인은 베지에스플라인에 비해 두가지 우점을 가진다. 첫째로, B스플라인다항식의 차수는 조종점들의 수에 무관계(일정한 제한을 가지고)하게 설정할수 있다. 둘째로, B스플라인은 스플라인곡선 또는 곡면의 형태를 국부조종할수 있다. 반면에 B스플라인은 베지에스플라인보다 더 복잡하다.

B스플라인곡선

B스플라인곡선우의 자리표위치들을 계산하는 일반적인 식은 혼합함수형식으로

$$\mathbf{P}(u)=\sum_{k=0}^{n}\mathbf{p}_k B_{k,d}(u),\quad u_{\min}\le u\le u_{\max},\quad 2\le d\le n+1 \qquad (10-54)$$

와 같이 쓸수 있다. 여기서 $\mathbf{p}_k$는 n+1개 입력조종점들의 모임이다. B스플라인의 이 형식화는 베지에스플라인에서의 형식화와 여러가지로 차이가 있다. 여기서는 파라메터 u의 범위가 B스플라인파라메터들을 어떻게 선택하는가에 관계된다. 그리고 B스플라인혼합함수 $B_{k,d}$는 d−1차의 다항식이다. 파라메터 d는 2부터 조종점들의 개수 n+1사이의 임의의 옹근수값으로 선택할수 있다(실제로 d의 값을 1로 설정할수도 있지만 이때 《곡선》은 조종점들의 점표시로 된다.). B스플라인에서 국부조종을 진행하기 위하여 u의 총체적인 범위의 부분구간들에서 혼합함수를 정의한다.

B스플라인곡선에 대한 혼합함수는 콕스-데보재귀공식

$$B_{k,1}(u)=\begin{cases}1, & u_k \le u < u_{k+1} \\ 0, & \text{그렇지 않을 때}\end{cases}$$
$$B_{k,d}(u)=\frac{u-u_k}{u_{k+d-1}-u_k}B_{k,d-1}(u)+\frac{u_{k+d}-u}{u_{k+d}-u_{k+1}}B_{k+1,d-1}(u) \quad (10-55)$$

로 정의한다. 여기서 매개 혼합함수는 u의 총체적인 범위안의 d개 부분구간들에서 정의된다. 선택된 부분구간의 끝점 u_j들의 모임을 매듭벡토르(knot vector)라고 한다. 부분구간의 끝점들에 대하여서는 관계 $u_j \le u_{j+1}$를 만족시키는 임의의 값을 선택할수 있다. 이때 u_{min}과 u_{max}에 대한 값은 선택하는 조종점들의 개수, 파라메터 d에 대하여 선택하는 값, 부분구간들(매듭벡토르)을 어떻게 설정하는가에 관계된다. 앞의 계산에서 분모가 값 0을 가지게 매듭벡토르의 요소들이 선택될수 있으므로 이 형식화에서는 0/0으로 평가되는 모든 항은 값 0을 가진다고 가정한다.

그림 10-41에서는 B스플라인의 국부조종특성을 보여 주었다. 국부조종외에 B스플라인에서는 다항식의 차수를 변화시킴이 없이 곡선설계에 리용하는 조종점들의 수를 변화시킬수 있으며 곡선의 형태를 조종하기 위하여 임의의 개수의 조종점들을 추가하거나 수정할수 있다. 류사하게 곡선을 쉽게 설계하기 위하여 매듭벡토르에서 값들의 개수를 증가시킬수 있다. 그러나 이렇게 할 때에는 매듭벡토르의 크기가 파라메터 n에 관계되므로 조종점들을 추가하여야 한다.

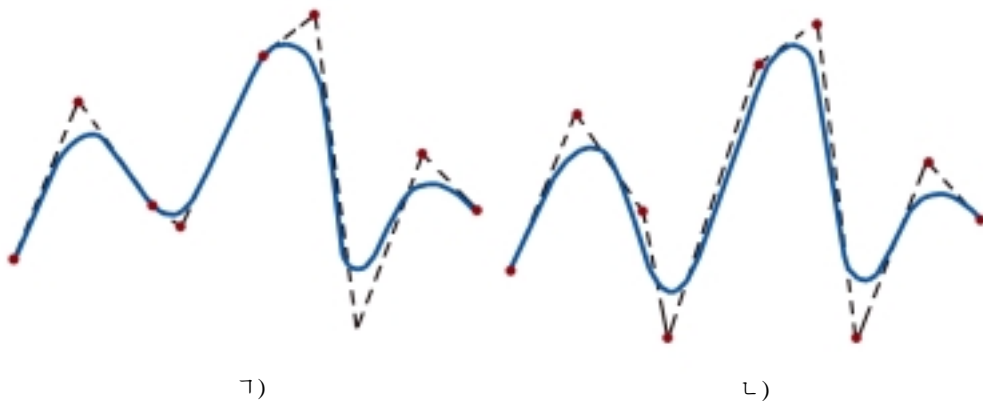



그림 10-41. B스플라인곡선의 국부적수정(ㄱ에서 1개 조종점을 변화시켜 곡선 ㄴ를 만들면 곡선은 변경된 조종점의 근방에서만 수정된다.)

B스플라인곡선은 다음과 같은 특성을 가진다.

- 다항식곡선의 차수는 d–1이며 다항식곡선은 u의 범위에서 C^{d-2}련속성을 가진다.
- n+1개 조종점에 대한 곡선은 n+1개 혼합함수들에 의해 표현된다.
- 매개 혼합함수 $B_{k,d}$는 매듭값 u_k에서 시작하여 u의 총체적인 범위의 d개 부분구간들에서 정의된다.
- 파라메터 u의 범위는 매듭벡토르에서 지적되는 $n+d+1$개의 값에 의하여 $n+d$개의 부분구간들로 나뉘여 진다.
- $\{u_0, u_1, \cdots, u_{n+d}\}$로 표시되는 매듭값들에 의하여 결과B스플라인곡선은 매듭값 u_{d-1}부터 u_{n+1}까지의 구간에서만 정의된다.
- 스플라인곡선의 매개 부분(두개의 련속한 매듭값사이)은 d개 조종점들의 영향을 받는다.
- 임의의 한개 조종점은 최대로 d개 부분곡선들의 형태에 영향을 미칠수 있다.

더우기 B스플라인곡선은 최대 d+1개의 조종점들의 볼록페포안에 있으며 따라서 B스플라인은 입력된 위치들에 팽팽하게 잡아 당겨 진다. 혼합함수들의 합은 매듭값 u_{d-1}부터 u_{n+1}사이의 구간에서 u의 임의의 값에 대하여 모두 1이다.

$$\sum_{k=0}^{n} B_{k,d}(u) = 1 \qquad (10-56)$$

조종점들의 위치와 파라메터 d의 값이 주어 지면 재귀관계식 10-55를 리용하여 혼합함수들을 얻기 위해 매듭값들을 지적할 필요가 있다. 매듭벡토르는 세가지 대체적인 부류 즉 균일, 열린균일, 비균일형으로 나눈다. 일반적으로 B스플라인은 선택하는 매듭벡토르의 부류에 따라 표현한다.

균일주기B스플라인

매듭값들사이의 간격이 일정할 때 결과곡선을 균일B스플라인(uniform B-spline)이라고 한다. 실례로 균일매듭벡토르를

{ -1.5, -1.0, -0.5, 0.0, 0.5, 1.0, 1.5, 2.0 }

과 같이 설정할수 있다. 매듭값들은 흔히

{ 0.0, 0.2, 0.4, 0.6, 0.8, 1.0 }

과 같이 0~1사이 범위로 정규화한다. 값 0에서 시작하여 간격 1을 가지고 균일매듭값들을 설정하면 대부분의 응용들에서 편리하다. 다음의 매듭벡토르는 이런 지적방법의 실례이다.

{ 0, 1, 2, 3, 4, 5, 6, 7 }

균일B스플라인은 주기적인 혼합함수들을 가진다. 즉 주어 진 값 n과 d에 대하여 모든 혼합함수들이 동일한 모양을 가진다. 매개 린접된 혼합함수는 단순히 앞에 놓인 함수가 옮겨 진것이다.

$$B_{k,d}(u) = B_{k+1,d}(u+\Delta u) = B_{k+2,d}(u+2\Delta u) \qquad (10-57)$$

여기서 Δu는 린접한 매듭값들사이의 간격이다. 그림 10-42에서는 다음의 실례에서 만들어 지는 4개의 조종점을 가지는 곡선에 대한 2차균일B스플라인혼합함수들을 보여 주었다.

실례 10-1. 2차균일B스플라인

균일옹근수매듭벡토르에 대한 B스플라인혼합함수들의 계산을 설명하기 위하여 파라메터값 d=n=3을 선택한다. 그러면 매듭벡토르는 n+d+1=7개의 매듭값을 포함해야 한다.

{ 0, 1, 2, 3, 4, 5, 6 }

파라메터 u의 범위는 0~6이고 n+d=6개의 부분구간이 있다.

4개의 매 혼합함수는 u의 총 범위의 d=3개의 부분구간에서 정의된다. 재귀관계식 10-55를 리용하여 첫 혼합함수를

$$B_{0,3}(u) = \begin{cases} \frac{1}{2}u^2, & 0 \le u < 1 \\ \frac{1}{2}u(2-u) + \frac{1}{2}(u-1)(3-u), & 1 \le u < 2 \\ \frac{1}{2}(3-u)^2, & 2 \le u < 3 \end{cases}$$

와 같이 얻는다. 다음번 주기혼합함수는 관계식 10-57을 리용하여 $B_{0,3}$에서 u를 u-1로 치환하고 시작위치를 하나 옮겨서 얻는다.

$$B_{1,3}(u)=\begin{cases}\frac{1}{2}(u-1)^2, & 1\le u<2\\ \frac{1}{2}(u-1)(3-u)+\frac{1}{2}(u-2)(4-u), & 2\le u<3\\ \frac{1}{2}(4-u)^2, & 3\le u<4\end{cases}$$

류사하게 나머지 두개의 주기함수들은 $B_{1,3}$을 오른쪽으로 련속 옮겨서 얻는다.

$$B_{2,3}(u)=\begin{cases}\frac{1}{2}(u-2)^2, & 2\le u<3\\ \frac{1}{2}(u-2)(4-u)+\frac{1}{2}(u-3)(5-u), & 3\le u<4\\ \frac{1}{2}(5-u)^2, & 4\le u<5\end{cases}$$

$$B_{3,3}(u)=\begin{cases}\frac{1}{2}(u-3)^2, & 3\le u<4\\ \frac{1}{2}(u-3)(5-u)+\frac{1}{2}(u-4)(6-u), & 4\le u<5\\ \frac{1}{2}(6-u)^2, & 5\le u<6\end{cases}$$

4개의 주기적인 2차혼합함수곡선을 그림 10-42에 주었다. 이것들은 B스플라인의 국부적인 특징을 보여 준다. 첫번째 조종점은 혼합함수 $B_{0,3}(u)$에 곱해 진다. 따라서 첫번째 조종점의 위치를 변화시키는 것은 곡선의 형태에 u=3까지만 영향을 미친다. 류사하게 마지막조종점은 $B_{3,3}$이 정의된 구간에서 스플라인곡선의 형태에 영향을 미친다.

그림 10-42에서는 이 실례에 대한 B스플라인곡선의 범위를 보여 주었다. u_{d-1}=2부터 u_{n+1}=4까지의 구간에는 모든 혼합함수들이 다 있다. 2아래와 4우에서는 모든 혼합함수들이 다 존재하지는 않는다. 이것이 다항식곡선의 범위이며 식 10-56에서 유효한 구간이다. 그러므로 모든 혼합함수들의 합은 이 구간에서 1이다. 이 구간밖에서는 모든 혼합함수들을 다 합할수 없다. 왜냐하면 2아래와 4우에서는 그것들이 모두 존재하는것이 아니기때문이다.

결과다항식곡선의 범위가 2부터 4까지이기때문에 곡선의 시작과 끝위치는 이 점들에서 혼합함수들을 계산하여

$$\mathbf{P}_{\text{start}}=\frac{1}{2}(\mathbf{p}_0+\mathbf{p}_1),\qquad \mathbf{P}_{\text{end}}=\frac{1}{2}(\mathbf{p}_2+\mathbf{p}_3)$$

과 같이 결정할수 있다. 그러므로 곡선은 첫 두 조종점사이의 중간위치에서 시작하고 마지막 두 조종점사이의 중간위치에서 끝난다.

또한 곡선의 시작과 끝위치에서 보조변수도함수들을 결정할수 있다. 혼합함수들의 도함수를 취하고 파라메터 u에 끝값들을 대입하면

$$\mathbf{P}'_{\text{start}}=\mathbf{p}_1-\mathbf{p}_0,\qquad \mathbf{P}'_{\text{end}}=\mathbf{p}_3-\mathbf{p}_2$$

을 얻는다. 시작위치에서 곡선의 보조변수경사도는 첫 두 조종점을 련결하는 선에 평행이며 곡선의 끝에서 보조변수경사도는 마지막 두 조종점을 련결하는 선에 평행이다.

xy평면에서 선택된 4개의 조종점들에 대한 2차주기B스플라인곡선의 실례를 그림 10-43에 보여주었다.

$B_{0,3}(u)$

ㄱ)

$B_{1,3}(u)$

ㄴ)

$B_{2,3}(u)$

ㄷ)

$B_{3,3}(u)$

ㄹ)

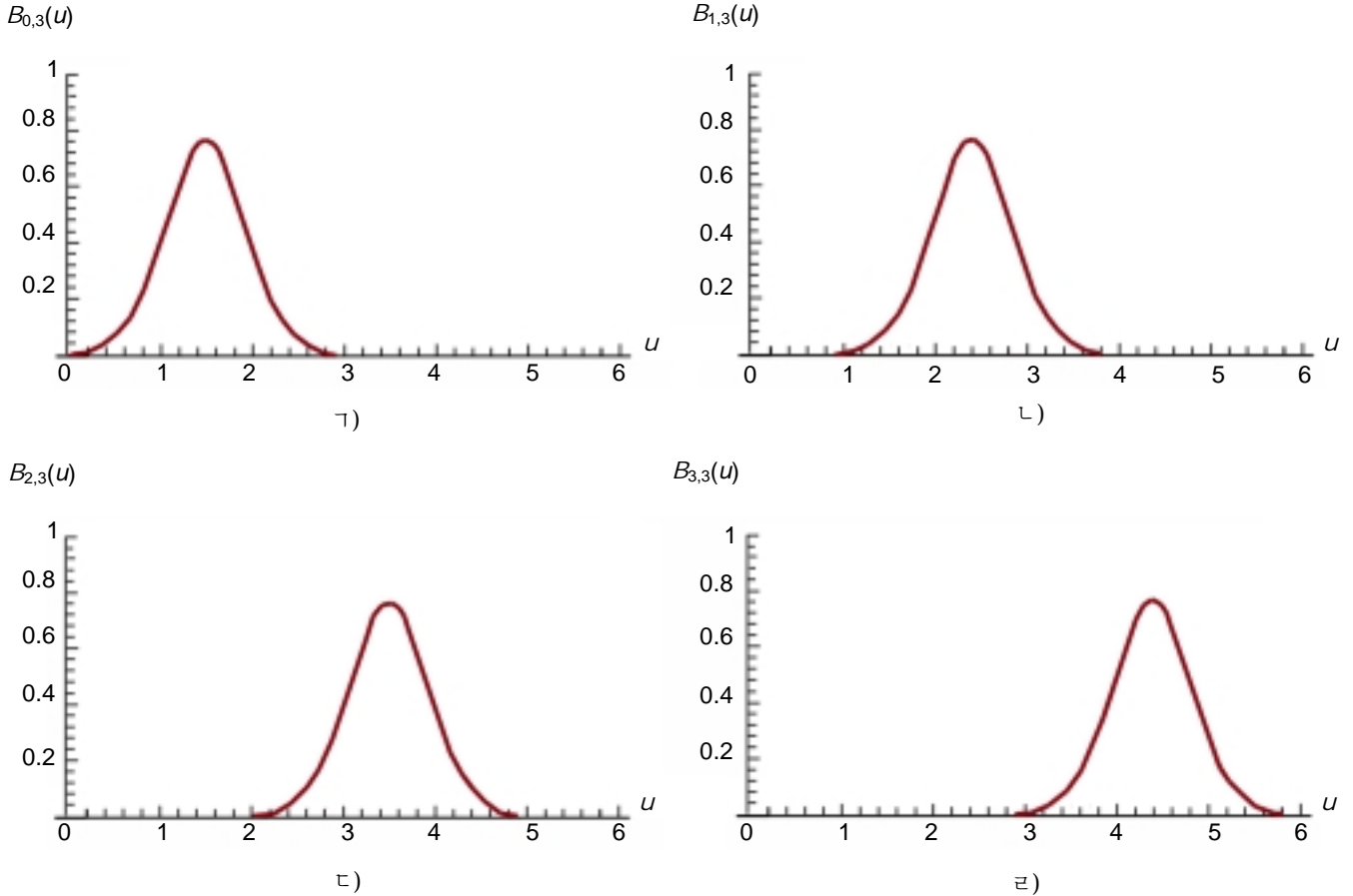



그림 10-42. $n=d=3$ 인 균일옹근수매듭벡토르에 대한 B스플라인의 주기적인 혼합함수들

앞의 실례에서 2차곡선은 첫 두 조종점사이에서 시작하고 마지막 두 조종점사이의 위치에서 끝난다는것을 주의하였다. 이 결과는 임의의 개수의 조종점들에 대하여 맞추어 지는 2차주기B스플라인인 경우에 유효하다. 일반적으로 더 높은 차수의 다항식에서 시작과 끝위치는 d-1개 조종점들의 무게붙은 평균이다. 어떤 조종점위치를 여러번 지적하여 스플라인곡선을 그 점에 더 가깝게 끌어 당길수 있다.

그림 10-43. xy평면의 4개의 조종점에 맞추어 지는 2차 주기B스플라인

파라메터 u가 0～1사이의 단위구간에로 넘어 가도록 혼합함수들을 다시 파라메터로 표시하여 주기B스플라인의 경계조건들에 대한 일반적인 표현을 얻을수 있다. 그러면 시작과 끝조건은 u=0과 u=1에서 얻어 진다.

3차주기B스플라인

3차주기B스플라인은 도형처리프로그람들에서 흔히 리용되기때문에 이 부류의 스플라인에 대한 형식화를 고찰한다. 주기스플라인은 특히 일정한 닫긴 곡선을 만드는데 쓸모 있다. 실례로 그림 10-44에서는 매 걸음에서 6개의 조종점들중 4개를 반복적으로 지적하는 방법으로 닫긴 곡선을 부분구간들에 의하여 만들수 있다. 임의의 련이은 3개의 조종점들이 동일하면 곡선은 그 자리표위치를 지난다.

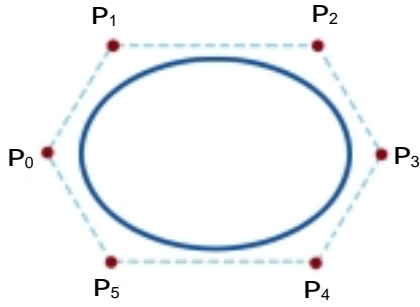



그림 10-44. 6개 조종점들을 주기적으로 지적하여 만들어 지는 닫긴 부분구간 3차주기B스플라인

3차곡선에서 d=4이고 매개 혼합함수는 u의 총 범위중 4개의 부분구간들에서 정의된다. 3차곡선을 4개의 조종점에 대하여 맞추려 한다면 앞의 소제목에서 2차주기B스플라인에 대하여 적용했던것과 마찬가지로 옹근수매듭벡토르

$$\{ 0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7 \}$$

과 재귀관계식 10-55를 주기적인 혼합함수들을 얻는데 리용할수 있다.

이 소제목에서는 3차주기B스플라인에 대한 또 다른 형식화를 고찰한다. 경계조건들을 가지고 시작하여 구간 $0 \le u \le 1$로 정규화된 혼합함수들을 얻는다. 이 형식화를 리용하여 또한 특성행렬을 쉽게 얻을수 있다. $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$으로 표식되는 련이은 4개의 조종점들을 가지는 3차주기B스플라인에 대한 경계조건은

$$\begin{aligned}
\mathbf{P}(0) &= \frac{1}{6}(\mathbf{p}_0 + 4\mathbf{p}_1 + \mathbf{p}_2) \\
\mathbf{P}(1) &= \frac{1}{6}(\mathbf{p}_1 + 4\mathbf{p}_2 + \mathbf{p}_3) \\
\mathbf{P}'(0) &= \frac{1}{2}(\mathbf{p}_2 - \mathbf{p}_0) \\
\mathbf{P}'(1) &= \frac{1}{2}(\mathbf{p}_3 - \mathbf{p}_1)
\end{aligned} \tag{10-58}$$

이다. 이 경계조건들은 기본스플라인에 대한것과 류사하다. 즉 부분곡선들은 4개의 조종점에 의하여 정의되며 매 부분곡선의 시작과 끝에서 보조변수도함수(경사도)는 린접한 조종점들을 련결하는 현에 평행이다. B스플라인부분곡선은 $\mathbf{p}_1$에 가까운 위치에서 시작하고 $\mathbf{p}_2$에 가까운 위치에서 끝난다.

4개의 조종점을 가지는 3차주기B스플라인에 대한 행렬형식화는

$$\mathbf{P}(u) = \begin{bmatrix} u^3 & u^3 & u & 1 \end{bmatrix} \cdot \mathbf{M}_B \cdot \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} \tag{10-59}$$

와 같이 쓸수 있다. 여기서 3차주기다항식에 대한 B스플라인행렬은

$$\mathbf{M}_B = \frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \tag{10-60}$$

이다. 이 행렬은 지적된 4개의 경계조건을 리용하여 일반3차다항식을 결수들에 대하여 풀어서 얻을수 있다.

또한 B스플라인방정식을 당김파라메터 t를 포함하도록 수정할수 있다(기본스플라인에서와 같다.). 당김을 가지는 3차주기B스플라인행렬은 형식

$$\mathbf{M}_{Bt}=\frac{1}{6}\begin{bmatrix}-t & 12-9t & 9t-12 & t\\ 3t & 12t-18 & 18-15t & 0\\ -3t & 0 & 3t & 0\\ t & 6-2t & t & 0\end{bmatrix} \qquad (10-61)$$

을 가진다. 이것은 t=1일 때 $\mathbf{M}_B$로 된다.

행렬표현을 다항식형식으로 전개하면 0~1사이의 파라메터범위에서 3차주기B스플라인에 대한 혼합함수들을 얻는다. 실례로 당김값 t=1일 때

$$\begin{aligned}B_{0,3}(u)&=\frac{1}{6}(1-u)^3, \qquad 0\le u\le 1\\ B_{1,3}(u)&=\frac{1}{6}(3u^3-6u^2+4)\\ B_{2,3}(u)&=\frac{1}{6}(-3u^3+3u^2+3u+1)\\ B_{3,3}(u)&=\frac{1}{6}u^3\end{aligned} \qquad (10-62)$$

이다.

열린균일B스플라인

이 부류의 B스플라인은 균일B스플라인과 비균일B스플라인사이에 있다. 때때로 그것은 특수한 형의 균일B스플라인으로 취급되기도 하고 때로는 비균일B스플라인부류에 있다고 고찰되기도 한다. 열린균일B스플라인(open uniform B-spline) 또는 간단히 열린스플라인에 대하여 매듭간격은 매듭값들이 d번 반복되는 끝을 제외하고는 일정하다.

다음의것은 열린균일옹근수매듭벡토르의 두가지 실례이며 매개는 값 0에서 시작한다.

{ 0, 0, 1, 2, 3, 3 } d=2, n=3

{ 0, 0, 0, 0, 1, 2, 2, 2, 2 } d=4, n=4

이 매듭벡토르들을 0~1사이의 단위구간에로 정규화할수 있다.

{ 0, 0, 0.33, 0.67, 1, 1 } d=2, n=3

{ 0, 0, 0, 0, 0.5, 1, 1, 1, 1 } d=4, n=4

파라메터 d와 n의 임의의 값에 대하여

$$u_j=\begin{cases}0, & 0\le j<d\\ j-d+1, & d\le j\le n\\ n-d+2, & j>n\end{cases} \qquad (10-63)$$

을 계산하면 옹근수값을 가지는 열린균일매듭벡토르를 만들수 있다. 여기서 j의 값은 0부터 $n+d$사이의 범위에 있다. 이 할당에 의하여 첫 d개의 매듭들에는 값 0이 할당되며 마지막 d개의 매듭들은 값 $n-d+2$를 가진다.

열린균일B스플라인은 베지에스플라인과 매우 류사한 특성을 가진다. 사실 $d=n+1$(다항식의 차수는 n이다.)일 때 열린B스플라인은 베지에스플라인으로 변하며 모든 매듭값들은 0 또는 1이다. 실례로 조종점의 개수가 4인 3차열린B스플라인(d=4)에서 매듭벡토르는

{ 0, 0, 0, 0, 1, 1, 1, 1 }

이다. 열린B스플라인에 대한 다항식곡선은 첫번째와 마지막조종점을 통과한다. 또한 첫번째 조종점에서 보조변수곡선의 경사도는 첫 두 조종점을 련결하는 선에 평행이다. 그리고 마지막조종점에서 보조변수경사도는 마지막 두 조종점을 련결하는 선에 평행이다. 그러므로 부분곡선들을 정합시키는 기하학적인 제한은 베지에곡선에서와 같다.

베지에곡선에서와 마찬가지로 동일한 자리표위치에서 여러개의 조종점들을 지적하면 임의의 B스플라인곡선을 그 위치에 더 가깝게 끌어 당긴다. 열린B스플라인이 첫번째 조종점에서 시작하고 마지막조종점에서 끝나기때문에 닫긴 곡선은 첫번째와 마지막조종점을 동일한 위치에서 지적하여 만든다.

실례 10-2. 2 차열린균일B스플라인

d=3이고 n=4(5개의 조종점)일 때 식 10-63으로부터 매듭벡토르에 대한 다음의 8개의 값을 얻는다.

$$\{u_0, u_1, u_2, u_3, u_4, u_5, u_6, u_7\} = \{0,\ 0,\ 0,\ 1,\ 2,\ 3,\ 3,\ 3\}$$

u의 총 범위는 7개의 부분구간들로 나뉘여 지며 5개의 혼합함수 $B_{k,3}$은 매듭위치 u_k에서 시작하여 3개의 부분구간들에서 정의된다. 그러므로 $B_{0,3}$은 u_0=0부터 u_3=1까지에서 정의되고 $B_{1,3}$은 u_1=0부터 u_4=2까지에서 정의되며 $B_{4,3}$은 u_4=2부터 u_7=3까지에서 정의된다. 재귀관계식 10-55로부터 혼합함수들에 대한 양함수적인 다항식은

$$B_{0,3}(u) = (1-u)^2, \qquad 0 \le u < 1$$

$$B_{1,3}(u) = \begin{cases} \frac{1}{2}u(4-3u), & 0 \le u < 1 \\ \frac{1}{2}(2-u)^2, & 1 \le u < 2 \end{cases}$$

$$B_{2,3}(u) = \begin{cases} \frac{1}{2}u^2, & 0 \le u < 1 \\ \frac{1}{2}u(2-u) + \frac{1}{2}(u-1)(3-u), & 1 \le u < 2 \\ \frac{1}{2}(3-u)^2, & 2 \le u < 3 \end{cases}$$

$$B_{3,3}(u) = \begin{cases} \frac{1}{2}(u-1)^2, & 1 \le u < 2 \\ \frac{1}{2}(3-u)(3u-5), & 2 \le u < 3 \end{cases}$$

$$B_{4,3}(u) = (u-2)^2, \qquad 2 \le u < 3$$

와 같이 얻어 진다. 그림 10-45에서는 이 5개의 혼합함수들의 형태를 보여 주었으며 B스플라인의 국부적인 특징을 다시 보여 준다. 혼합함수 $B_{0,3}$은 0~1사이의 부분구간에서만 0이 아니며 따라서 첫번째 조종점은 이 구간에서만 곡선에 영향을 미친다. 류사하게 함수 $B_{4,3}$은 2~3사이의 구간밖에서는 0이며 마지막조종점의 위치는 곡선의 시작과 중간부분의 형태에 영향을 미치지 않는다.

열린B스플라인에 대한 행렬형식화는 주기균일B스플라인에서보다 편리하게 만들어 지지 않는다. 이것은 매듭벡토르의 시작과 끝에서 매듭값들이 중복되기때문이다.

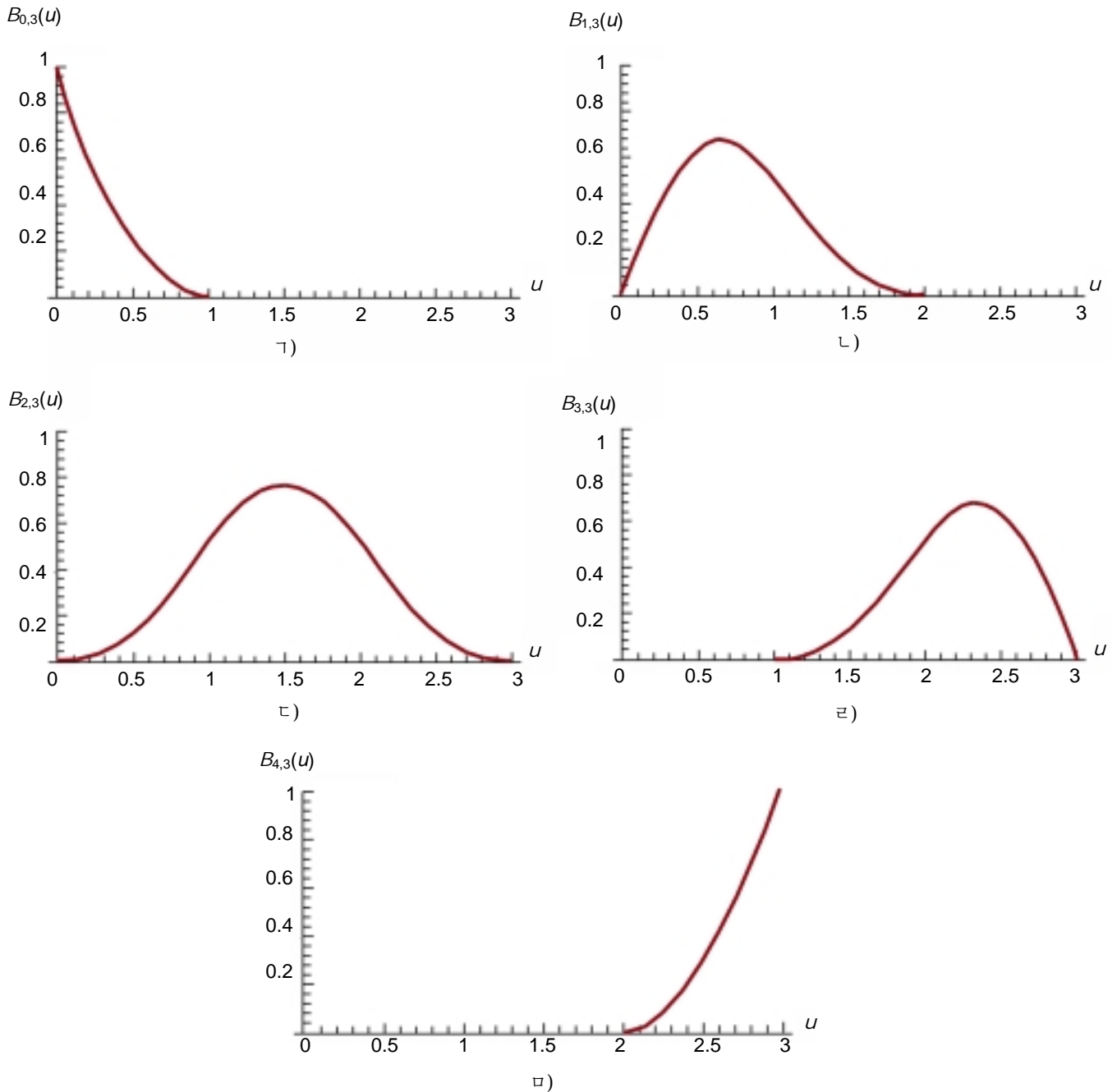



그림 10-45. n=4, d=3일 때 열린균일B스플라인의 혼합함수

비균일B스플라인

이 부류의 스플라인에서 매듭벡토르에 대하여 임의의 값과 간격을 지적할수 있다. 비균일B스플라인(nonuniform B-spline)에서는 내부매듭값들이 반복되거나 매듭값들사이가 같지 않게 선택할수 있다. 몇가지 실례는

{ 0, 1, 2, 3, 3, 4 }

{ 0, 2, 2, 3, 3, 6 }

{ 0, 0, 0, 1, 1, 3, 3, 3 }

{ 0, 0.2, 0.6, 0.9, 1.0 }

이다.

비균일B스플라인은 곡선의 형태를 보다 적응적으로 조종할수 있게 한다. 매듭벡토르에서 구간간격이 균일하지 않으므로 서로 다른 구간에서 서로 다른 형태의 혼합함수를 얻으며 그것은 스플라인형태를 조정하는데 리용될수 있다. 매듭에 반복을 많이 주면 곡선형태에서 미묘한 변화와 불련속적인 모양까지도 얻을수 있다. 매듭값을 반복하면 매 반복에서 련속성이 1씩 줄어 들게 된다.

비균일B스플라인의 혼합함수들은 균일 및 열린B스플라인에서 설명한것과 류사한 방법으로 얻는다. n+1개의 조종점들의 모임이 주어 지면 다항식의 차수를 설정하고 매듭값들을 선택한다. 다음에 재귀관계를 리용하여 혼합함수들의 모임을 얻거나 또는 곡선을 현시하기 위하여 직접 곡선우의 위치들을 계산할수 있다. 도형처리프로그람들은 계산을 줄이기 위하여 자주 매듭구간을 0~1이 되도록 제한한다. 다음에 특성행렬들을 기억하고 그리려는 곡선의 매 점에 대하여 재귀관계를 평가함이 없이 스플라인곡선을 따라 값들을 계산하는데 특성행렬들을 리용한다.

B스플라인곡면

B스플라인곡면의 형식화는 베지에스플라인에서와 류사하다. 형식

$$\mathbf{P}(u,v)=\sum_{k_1=0}^{n1}\sum_{k_2=0}^{n2}\mathbf{p}_{k_1,k_2}B_{k_1,d_1}(u)B_{k_2,d_2}(v) \qquad (10-64)$$

의 B스플라인혼합함수들의 적을 리용하여 B스플라인곡면에서의 벡토르점함수를 얻을수 있다. 여기서 $\mathbf{p}_{k1,k2}$은 $(n_1+1)\times(n_2+1)$개 조종점들의 위치를 지적한다.

B스플라인곡면은 그의 성분B스플라인곡선과 같은 특성을 보여 준다. 곡면은 파라메터 d_1과 d_2(그것들은 리용할 다항식의 차수를 결정한다.)에 대하여 선택되는 값과 지적된 매듭벡토르로부터 만들어 질수 있다. 그림 10-46에서는 B스플라인곡면으로 모형화된 물체를 보여 주었다.

그림 10-46. SOFTIMAGE,Inc., Montreal의 Daniel Langlois에 의하여 설계되고 모형화된 180,000개의 B스플라인곡면쪼각들을 리용하는 원형직승기(이 장면은 다음에 광선추적법, 곱보만들기, 반사넘기기를 리용하여 실감처리하였다.)

10절. 베타스플라인

B스플라인의 일반화는 베타스플라인(beta-spline, β-spline)이며 그것은 1계 및 2계보조변수도함수에 기하학적련속성조건들을 부과하여 형식화한다. 베타스플라인에 대한 련속성파라메터들을 β파라메터라고 한다.

베타스플라인련속성조건

지적된 매듭벡토르에 대하여 개개 매듭 u_j의 왼쪽과 오른쪽의 스플라인부분들을 위치벡토르

$\mathbf{P}_{j-1}(u)$와 $\mathbf{P}_j(u)$로 지적할수 있다(그림10-47). u_j에서 0계련속성(위치련속성) G^0은

$$\mathbf{P}_{j-1}(u_j) = \mathbf{P}_j(u_j) \qquad (10-65)$$

에 의하여 얻는다.

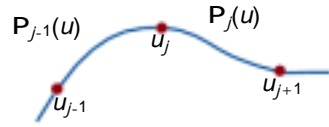



그림 10-47. 매듭 u_j의 왼쪽과 오른쪽의 부분곡선에 대한 위치벡토르

1계련속성(단위접선련속성) G^1은 비례하는 접선벡토르들에 의하여 얻는다.

$$\beta_1 \mathbf{P}'_{j-1}(u_j) = \mathbf{P}'_j(u_j), \qquad \beta_1 > 0 \qquad (10-66)$$

여기서 보조변수1계도함수들은 비례하며 단위접선벡토르들은 매듭을 가로질러서 련속이다.

2계련속성(곡률벡토르련속성) G^2은 조건

$$\beta_1^2 \mathbf{P}''_{j-1}(u_j) + \beta_2 \mathbf{P}'_{j-1}(u_j) = \mathbf{P}''_j(u_j) \qquad (10-67)$$

에 의하여 얻는다. 여기서 β_2에는 임의의 실수를 줄수 있으며 $\beta_1>0$이다. 곡률벡토르는 곡선이 위치 u_j에서 구부러 지는 량을 측정한다. $\beta_1=1$, $\beta_2=0$일 때 베타스플라인은 B스플라인으로 변한다.

파라메터 β_1은 곡선의 비대칭도를 조종하기때문에 편위파라메터라고 한다. $\beta_1>1$일 때 곡선은 매듭에서 오른쪽으로 단위접선벡토르의 방향으로 팽팽하게 되는 경향이 있다. $0<\beta_1<1$일 때 곡선은 왼쪽으로 팽팽하게 되는 경향이 있다. 스플라인곡선의 형태에 미치는 β_1의 효과를 그림 10-48에서 보여주었다.

파라메터 β_2는 스플라인이 얼마나 팽팽하게 또는 느슨하게 조종그라프를 맞추는가를 조종하기때문에 당김파라메터라고 한다. 그림 10-49에 보여 준바와 같이 β_2이 증가할 때 곡선은 조종그라프의 형태에 접근한다.

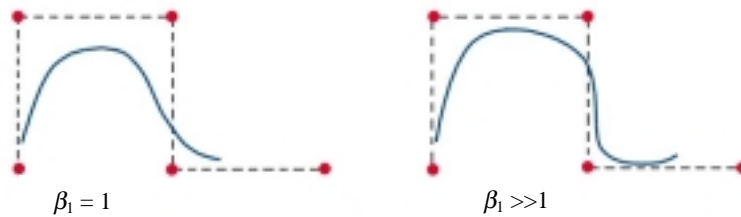



그림 10-48. 베타스플라인곡선의 형태에 미치는 파라메터 β_1의 효과

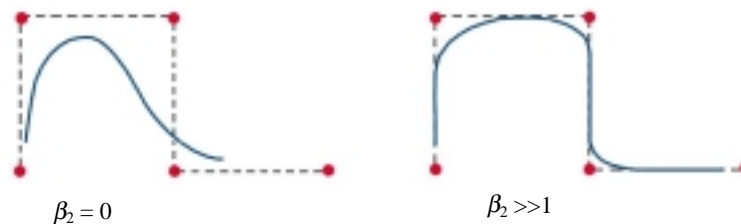



그림 10-49. 베타스플라인곡선의 형태에 미치는 파라메터 β_2의 효과

3 차주기베타스플라인의 행렬표현

베타스플라인의 경계조건들을 균일매듭벡토르를 가지는 3차다항식에 적용하면 주기베타스플라인에 대한 다음의 행렬표현을 얻는다.

$$\mathbf{M}_\beta = \frac{1}{\delta}\begin{bmatrix} -2\beta_1^3 & 2(\beta_2+\beta_1^3+\beta_1^2+\beta_1) & -2(\beta_2+\beta_1^2+\beta_1+1) & 2 \\ 6\beta_1^3 & -3(\beta_2+2\beta_1^3+2\beta_1^2) & 3(\beta_2+2\beta_1^2) & 0 \\ -6\beta_1^3 & 6(\beta_1^3-\beta_1) & 6\beta_1 & 0 \\ 2\beta_1^3 & \beta_2+4(\beta_1^2+\beta_1) & 2 & 0 \end{bmatrix} \qquad (10-68)$$

여기서 $\delta = \beta_2 + 2\beta_1^3 + 4\beta_1^2 + 4\beta_1 + 2$ 이다.

β_1=1, β_2=0일 때 B스플라인행렬 $\mathbf{M}_B$를 얻는다. 그리고

$$\beta_1 = 1, \qquad \beta_2 = \frac{12}{t}(1-t)$$

일 때 당김을 가지는 B스플라인행렬 $\mathbf{M}_{Bt}$를 얻는다.

11절. 유리스플라인

유리함수는 두 다항식의 비이다. 그러므로 유리스플라인(rational spline)은 두 스플라인함수의 비이다. 실례로 유리B스플라인곡선은 다음의 위치벡토르에 의하여 표현할수 있다.

$$\mathbf{P}(u) = \frac{\sum_{k=0}^{n} \omega_k \mathbf{p}_k B_{k,d}(u)}{\sum_{k=0}^{n} \omega_k B_{k,d}(u)} \qquad (10-69)$$

여기서 $\mathbf{p}_k$는 n+1개 조종점위치들의 모임이다. 파라메터 ω_k는 조종점의 무게결수이다. ω_k의 값이 클수록 곡선은 그 무게결수를 가지는 조종점 $\mathbf{p}_k$쪽으로 더 가깝게 당겨 진다. 모든 무게결수들이 값 1로 설정될 때 식 10-69의 분모는 1(혼합함수들의 합)이기때문에 표준B스플라인곡선을 얻는다.

유리스플라인은 비유리스플라인에 비하여 두가지 중요한 우점을 가진다. 첫째로, 유리스플라인은 원 및 타원과 같은 2차곡선(원추곡선)에 대한 정확한 표현을 제공한다. 다항식인 비유리스플라인은 원추곡선을 근사만 할수 있다. 이것은 도형처리프로그람들이 서로 다른 형태들을 처리할 때 곡선함수들의 서고가 없이 모든 곡선형태들을 하나의 표현 즉 유리스플라인으로 모형화할수 있게 한다. 유리스플라인의 우점은 둘째로, 그것이 원근보기변환(12장 3절)에 대하여 불변이라는것이다. 이것은 유리곡선의 조종점들에 원근보기변환을 적용할수 있으며 정확한 상을 얻을수 있다는것을 의미한다. 한편 비유리스플라인은 원근보기변환에 대하여 불변이 아니다. 일반적으로 도형처리설계프로그람들은 유리B스플라인을 만드는데 비균일매듭벡토르표현을 리용한다. 이 스플라인을 비균일유리B스플라인(NURBs, nonuniform rational B-spline)이라고 한다. 유리스플라인에 대하여 동차자리표표현을 리용한다. 왜냐하면 분모는 조종점들의 4차원표현에서 동차결수로 취급할수 있기때문이다. 그러므로 유리스플라인은 4차원비유리스플라인의 3차원공간에로의 투영이라고 생각할수 있다.

유리B스플라인표현을 만드는것은 비유리표현을 만드는것과 같은 절차로 수행한다. 조종점들의 모임, 다항식의 차수, 무게결수, 매듭벡토르가 주어 지면 혼합함수들을 얻기 위하여 재귀관계를 적용한다.

원추곡선을 NURBs로 표시하기 위하여 2차스플라인함수(d=3)와 3개의 조종점을 리용할수 있다.

이것은 열린매듭벡토르

$$\{\ 0,\ 0,\ 0,\ 1,\ 1,\ 1\ \}$$

에 의하여 정의되는 B스플라인함수로 할수 있는데 그것은 2차베지에스플라인과 같다. 무게함수들은 다음의 값으로 설정한다.

$$\omega_0 = \omega_2 = 1$$
$$\omega_1 = \frac{r}{1-r}, \qquad 0 \le r < 1 \tag{10-70}$$

유리B스플라인표현은

$$\mathbf{P}(u) = \frac{\mathbf{p}_0 B_{0,3}(u) + [r/(1-r)]\mathbf{p}_1 B_{1,3}(u) + \mathbf{p}_2 B_{2,3}(u)}{B_{0,3}(u) + [r/(1-r)]B_{1,3}(u) + B_{2,3}(u)} \tag{10-71}$$

이다. 그러면 다음의 파라메터 r값들에 의해 여러가지 원추곡선(그림 10-50)을 얻는다.

$r>1/2$, $\omega_1>1$ (부분쌍곡선)
$r=1/2$, $\omega_1=1$ (부분포물선)
$r<1/2$, $\omega_1<1$ (부분타원)
$r=0$, $\omega_1=0$ (선분)

$\omega_1=\cos\phi$로 설정하고 조종점들을

$$\mathbf{p}_0 = (0, 1), \qquad \mathbf{p}_1 = (1, 1), \qquad \mathbf{p}_2 = (1, 0)$$

과 같이 선택하여 xy평면의 첫 4분구에서 단위원의 1/4호를 만들수 있다(그림 10-51). 단위원의 다른 부분들은 서로 다른 조종점위치에 의하여 얻을수 있다. 옹근원은 평면에서 기하학적인 변환을 리용하여 만들수 있다. 실례로 다른 3개 4분구의 원호를 만들기 위하여 1/4원호를 x 및 y축에 대하여 반사시킬수 있다.

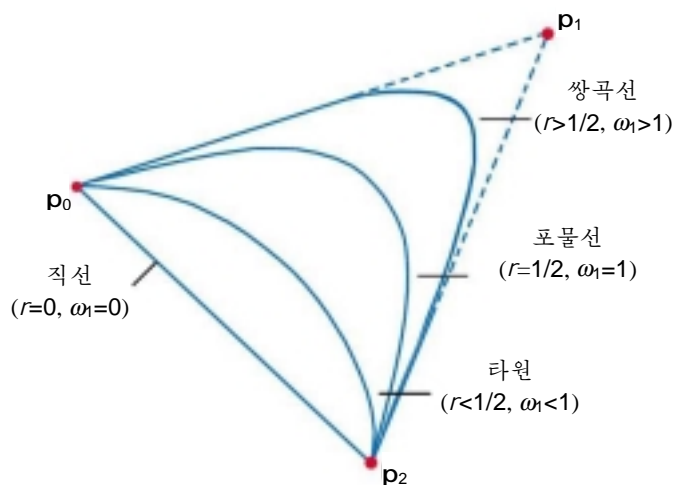



그림 10-50. 유리스플라인의 무게결수 ω_1의 여러가지 값에 의하여 얻어 지는 원추곡선들

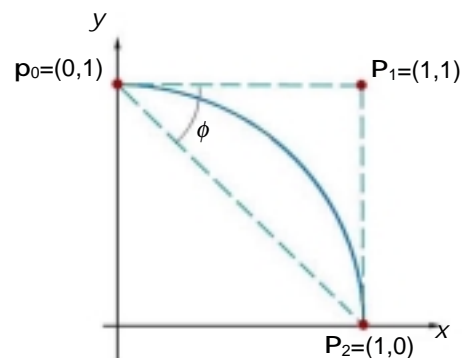



그림 10-51. xy 평면의 첫번째 4 분구의 원호

일부 CAD체계들에서 원추곡선은 한 호의 3개 점을 지적하여 만든다. 그러면 유리동차자리표스플라인표현은 선택된 원추곡선을 만드는 조종점위치들을 계산하여 결정한다. 실례로 *xy*평면에서 첫번째 4분구의 단위원호에 대한 동차표현은

$$\begin{bmatrix} x_h(u) \\ y_h(u) \\ z_h(u) \\ h \end{bmatrix} = \begin{bmatrix} 1-u^2 \\ 2u \\ 0 \\ 1+u^2 \end{bmatrix}$$

이다.

12절. 스플라인표현들사이의 변환

때때로 한 스플라인표현으로부터 다른데로 전환할수 있다. 실례로 베지에표현은 스플라인곡선을 부분분할하는 제일 편리한 표현이며 한편 B스플라인표현은 설계에서 아주 편리하다. 그러므로 곡선은 B스플라인부분들을 리용하여 설계한다. 다음에 물체를 현시하기 위하여 곡선을 따라 자리표위치를 정하는 재귀적인 부분분할절차를 써서 등가베지에표현으로 전환할수 있다.

다음의 행렬적

$$\mathbf{P}(u) = \mathbf{U} \cdot \mathbf{M}_{\text{spline1}} \cdot \mathbf{M}_{\text{geom1}} \tag{10-72}$$

으로 표현할수 있는 물체의 스플라인표현을 가진다고 하자. 여기서 $\mathbf{M}_{\text{spline1}}$은 이 스플라인의 특성행렬, $\mathbf{M}_{\text{geom1}}$는 기타 제한(실례로 조종점자리표)들의 렬벡토르이다. 스플라인행렬 $\mathbf{M}_{\text{spline2}}$을 가지는 두번째 표현으로 변환하기 위하여 물체에 대한 동일한 벡토르점함수를 만드는 기하제한행렬 $\mathbf{M}_{\text{geom2}}$을 결정하여야 한다. 즉

$$\mathbf{P}(u) = \mathbf{U} \cdot \mathbf{M}_{\text{spline2}} \cdot \mathbf{M}_{\text{geom2}} \tag{10-73}$$

또는

$$\mathbf{U} \cdot \mathbf{M}_{\text{spline2}} \cdot \mathbf{M}_{\text{geom2}} = \mathbf{U} \cdot \mathbf{M}_{\text{spline1}} \cdot \mathbf{M}_{\text{geoml}}$$

$\mathbf{M}_{\text{geom2}}$에 대하여 풀면

$$\begin{aligned} \mathbf{M}_{\text{geom2}} &= \mathbf{M}_{\text{spline2}}^{-1} \cdot \mathbf{M}_{\text{spline1}} \cdot \mathbf{M}_{\text{geom1}} \\ &= \mathbf{M}_{\text{s1,s2}} \cdot \mathbf{M}_{\text{geom1}} \end{aligned} \tag{10-74}$$

이며 스플라인의 첫번째 표현을 두번째 표현으로 변환하는 변환행렬은

$$\mathbf{M}_{\text{s1,s2}} = \mathbf{M}_{\text{spline2}}^{-1} \cdot \mathbf{M}_{\text{spline1}} \tag{10-75}$$

과 같이 계산된다.

비균일B스플라인은 일반적인 행렬에 의하여 특징 지을수 있다. 그러나 비균일B스플라인을 베지에표현으로 변화시키기 위하여 매듭순서렬을 다시 정렬시킬수 있다. 그러면 베지에행렬은 임의의 다른 형식으로 변환될수 있다.

다음의 실례는 주기3차B스플라인표현으로부터 3차베지에스플라인표현으로 변환하기 위한 변환행렬을 계산한다.

$$\mathbf{M}_{B,\mathrm{Bez}} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}^{-1} \cdot \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} = \frac{1}{6} \begin{bmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{bmatrix} \quad (10-76)$$

그리고 3차베지에표현으로부터 주기3차B스플라인표현으로 변환하기 위한 변환행렬은

$$\mathbf{M}_{\mathrm{Bez},B} = \begin{bmatrix} -1/6 & 1/2 & -1/2 & 1/6 \\ 1/2 & -1 & 1/2 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 1/6 & 2/3 & 1/6 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 6 & -7 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -7 & 6 \end{bmatrix} \quad (10-77)$$

13절. 스플라인곡선과 곡면의 현시

스플라인곡선 또는 곡면을 현시하기 위하여 현시장치의 화소위치로 투영하는 곡선 또는 곡면의 자리표위치들을 결정하여야 한다. 이것은 보조변수다항식스플라인의 자리표위치들을 일정한 증분으로 평가하여야 한다는것을 의미한다. 스플라인곡선 또는 곡면의 자리표위치들을 계산하는데 리용할 수 있는 여러가지 방법들이 있다.

호너규칙

다항식의 매 항의 련속적인 기계적계산과는 다른 다항식을 평가하는 제일 간단한 방법은 호너규칙이며 그것은 련속인수분해를 리용하여 계산을 진행한다. 이것은 매 걸음에서 하나의 곱하기와 하나의 더하기를 요구한다. 차수가 n인 다항식인 경우 n개 걸음이 있다.

실례로 자리표위치가

$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x \quad (10-78)$$

와 같이 표현되는 3차스플라인표현을 가진다고 하자. y 및 z에 대한 표현도 류사하다. 파라메터 u의 개개 값에 대하여 이 다항식을 다음의 인수분해순서로 계산한다.

$$x(u) = [(a_x u + b_x)u + c_x]u + d_x \quad (10-79)$$

매 x값의 계산은 세개의 곱하기와 세개의 더하기를 요구하며 따라서 3차스플라인곡선을 따라 매 자리표위치 (x, y, z)를 결정하는것은 9개의 곱하기와 9개의 더하기를 요구한다.

특히 더 높은 차수의 다항식(3보다 더 큰 차수)들에 대한 추가적인 인수분해기교는 호너방법에

서 요구되는 계산수를 줄이는데 적용될수 있다. 그러나 스플라인의 자리표위치들의 반복결정은 앞계차계산 또는 스플라인부분분할방법을 리용하여 훨씬 더 빨리 할수 있다.

앞계차계산

다항식함수를 평가하는 빠른 방법은 앞에서 계산된 값을 실례로

$$x_{k+1} = x_k + \Delta x_k \tag{10-80}$$

와 같이 증가시켜 련속적인 값들을 재귀적으로 발생시키는것이다. 그러므로 증분과 어떤 걸음에서의 x_k의 값을 알면 다음값을 그 걸음에서의 x_k값에 증분을 더하여 얻는다. 매 걸음에서의 증분 Δx_k를 앞계차라고 한다. u의 총 범위를 고정크기 δ의 부분구간들로 나누면 두개의 련속인 x위치는 x_k=$x(u_k)$와 x_{k+1}= $x(u_{k+1})$에 있다. 여기서

$$u_{k+1} = u_k + \delta, \qquad k = 0, 1, 2, \cdots \tag{10-81}$$

이고 u_0=0이다.

이 방법을 설명하기 위하여 선형스플라인표현 $x(u)$=$a_x u$+b_x를 가진다고 하자 . 두개의 련속인 x자리표위치들은

$$\begin{aligned} x_k &= a_x u_k + b_x \\ x_{k+1} &= a_x(u_k + \delta) + b_x \end{aligned} \tag{10-82}$$

와 같이 표현된다. 두 방정식을 덜면 앞계차 Δx_k=$a_x\delta$를 얻는다. 이 경우 앞계차는 상수이다. 더 높은 차수의 다항식들에서 앞계차는 그자체가 본래의 다항식보다 하나 작은 차수를 가지는 파라메터 u의 다항식함수이다.

식 10-78의 3차스플라인표현에서 두개의 련속인 x자리표위치들은 다항식표현

$$\begin{aligned} x_k &= a_x u_k^3 + b_x u_k^2 + c_x u_k + d_x \\ x_{k+1} &= a_x(u_k + \delta)^3 + b_x(u_k + \delta)^2 + c_x(u_k + \delta) + d_x \end{aligned} \tag{10-83}$$

을 가진다. 앞계차는

$$\Delta x_k = 3a_x\delta u_k^2 + (3a_x\delta^2 + 2b_x\delta)u_k + (a_x\delta^3 + b_x\delta^2 + c_x\delta) \tag{10-84}$$

로 평가된다. 그것은 파라메터 u_k의 2차함수이다. Δx_k는 u의 다항식함수이기때문에 Δx_k의 련속적인 값들을 얻기 위하여 동일한 증가절차를 리용할수 있다. 즉

$$\Delta x_{k+1} = \Delta x_k + \Delta^2 x_k \tag{10-85}$$

여기서 두번째 앞계차는 선형함수이다.

$$\Delta^2 x_k = 6a_x\delta^2 u_k + 6a_x\delta^3 + 2b_x\delta^2 \tag{10-86}$$

이 처리를 한번 더 반복하면

$$\Delta^2 x_{k+1} = \Delta^2 x_k + \Delta^3 x_k \tag{10-87}$$

로 쓸수 있다. 여기서 세번째 앞계차는 상수이다.

$$\Delta^3 x_k = 6a_x\delta^3 \tag{10-88}$$

식 10-80, 10-85, 10-87, 10-88은 3차곡선을 따라 점들의 증가앞계차계산을 제공한다. 걸음크기 δ를 가지고 u_0=0에서 시작하면 x자리표에 대한 초기값과 그의 첫 두 앞계차를

$$\begin{aligned} x_0 &= d_x \\ \Delta x_0 &= a_x\delta^3 + b_x\delta^2 + c_x\delta \\ \Delta^2 x_0 &= 6a_x\delta^3 + 2b_x\delta^2 \end{aligned} \tag{10-89}$$

와 같이 얻는다. 이 초기값들이 계산되면 매 련속한 x자리표위치에 대한 계산은 3개의 더하기만을 요구한다.

앞계차방법을 임의의 차수 n의 스플라인곡선을 따라 위치들을 결정하는데 적용할수 있다. 매개 련속인 자리표위치 (x, y, z)는 $3n$개 더하기의 련속으로 평가된다. 곡면에 대하여 증가계산은 파라메터 u와 v에 다같이 적용된다.

부분분할방법

재귀적인 스플라인부분분할절차는 주어 진 부분곡선을 절반으로 반복 나누는데 적용되며 매 걸음에서 조종점들의 수를 증가시킨다. 부분분할방법은 근사스플라인곡선의 현시에 쓸모 있다. 왜냐하면 조종그라프가 곡선경로에 근사할 때까지 부분분할처리를 계속할수 있기때문이다. 그러면 조종점의 자리표들은 곡선위치로 표시될수 있다. 부분분할의 다른 응용은 곡선을 만드는데 더 많은 조종점들을 만드는것이다. 그러므로 일반적인 곡선형태를 몇개의 조종점들로 설계하고 다음에 추가적인 조종점들을 얻기 위하여 부분분할절차를 적용할수 있다. 추가적인 조종점들에 의하여 곡선의 작은 부분에 대한 미세한 조종을 할수 있다.

스플라인부분분할은 베지에곡선부분에 제일 쉽게 적용된다. 왜냐하면 그 곡선은 첫번째와 마지막조종점을 통과하고 파라메터 u가 항상 0과 1사이의 범위에 있으며 조종점들이 곡선경로에 《충분히 가까운》때를 쉽게 결정할수 있기때문이다. 베지에부분분할은 다음의 조작렬에 의하여 다른 스플라인표현에 적용될수 있다.

1. 사용하는 스플라인표현을 베지에표현으로 변환한다.
2. 베지에부분분할알고리듬을 적용한다.
3. 베지에표현을 본래의 스플라인표현에로 다시 변환한다.

그림 10-52에서는 3차베지에부분곡선의 재귀적인 부분분할에서 첫 걸음을 보여 주었다. 베지에곡선우의 위치들은 보조변수점함수 $\mathbf{P}(u)$, $0 \le u \le 1$에 의하여 표현된다. 첫번째 부분분할걸음에서 본래의 곡선을 두 부분으로 나누기 위하여 중간점 $\mathbf{P}(0.5)$를 리용한다. 그러면 첫번째 부분은 점함수 $\mathbf{P}_1(s)$로 표현되고 두번째 부분은 $\mathbf{P}_2(t)$로 표현된다. 여기서

$$\begin{aligned} s &= 2u, & 0 \le u \le 0.5 \\ t &= 2u-1, & 0.5 \le u \le 1 \end{aligned} \qquad (10-90)$$

두개의 새 부분곡선들은 매개가 본래의 부분곡선과 같은 수의 조종점을 가진다. 또한 매개 새 부분곡선의 두 끝에서 경계조건(위치와 보조변수경사도)은 본래의 곡선 $\mathbf{P}(u)$에 대한 위치 및 경사도값과 정합하여야 한다. 이것은 매 부분곡선에 대하여 조종점위치들을 결정하는데 리용할수 있는 4개의 조건을 준다. 곡선의 첫 절반에 대하여 4개의 새 조종점들은

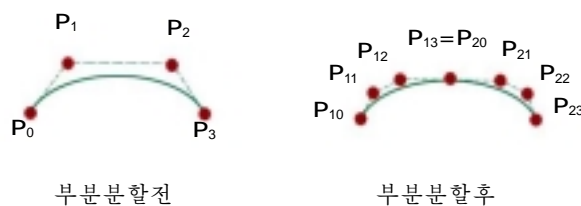



그림 10-52. 3차베지에부분곡선을 매개가 4개의 조종점을 가지는 두개의 부분으로 부분분할

$$\begin{aligned}\mathbf{p}_{1,0} &= \mathbf{p}_0\\ \mathbf{p}_{1,1} &= \frac{1}{2}(\mathbf{p}_0+\mathbf{p}_1)\\ \mathbf{p}_{1,2} &= \frac{1}{4}(\mathbf{p}_0+2\mathbf{p}_1+\mathbf{p}_2)\\ \mathbf{p}_{1,3} &= \frac{1}{8}(\mathbf{p}_0+3\mathbf{p}_1+3\mathbf{p}_2+\mathbf{p}_3)\end{aligned} \tag{10-91}$$

이다. 그리고 곡선의 두번째 절반에 대하여 4개의 조종점

$$\begin{aligned}\mathbf{p}_{2,0} &= \frac{1}{8}(\mathbf{p}_0+3\mathbf{p}_1+3\mathbf{p}_2+\mathbf{p}_3)\\ \mathbf{p}_{2,1} &= \frac{1}{4}(\mathbf{p}_1+2\mathbf{p}_2+\mathbf{p}_3)\\ \mathbf{p}_{2,2} &= \frac{1}{2}(\mathbf{p}_2+\mathbf{p}_3)\\ \mathbf{p}_{2,3} &= \mathbf{p}_3\end{aligned} \tag{10-92}$$

을 얻는다. 새 조종점들을 계산하는 능률적인 순서는

$$\begin{aligned}\mathbf{p}_{1,0} &= \mathbf{p}_0\\ \mathbf{p}_{1,1} &= \frac{1}{2}(\mathbf{p}_0+\mathbf{p}_1)\\ \mathbf{T} &= \frac{1}{2}(\mathbf{p}_1+\mathbf{p}_2)\\ \mathbf{p}_{1,2} &= \frac{1}{2}(\mathbf{p}_{1,1}+\mathbf{T})\\ \mathbf{p}_{2,3} &= \mathbf{p}_3\\ \mathbf{p}_{2,2} &= \frac{1}{2}(\mathbf{p}_2+\mathbf{p}_3)\\ \mathbf{p}_{2,1} &= \frac{1}{2}(\mathbf{T}+\mathbf{p}_{2,2})\\ \mathbf{p}_{2,0} &= \frac{1}{2}(\mathbf{p}_{1,2}+\mathbf{p}_{2,1})\\ \mathbf{p}_{1,3} &= \mathbf{p}_{2,0}\end{aligned} \tag{10-93}$$

와 같이 더하기와 밀기(2로 나누기)조작만으로 설정할수 있다.

이 걸음들은 더 많은 조종점들을 얻기 위하여 곡선을 부분분할하는가 아니면 근사곡선위치들을 정하는가에 따라 임의의 회수만큼 반복할수 있다. 현시점들의 모임을 얻기 위하여 부분분할할 때 부분곡선이 충분히 작으면 부분분할절차를 끝낼수 있다. 이것을 결정하는 한가지 방법은 매 부분에 대하여 첫번째 조종점으로부터 마지막조종점까지의 거리를 검사하는것이다. 이 거리가 충분히 작으면 부분분할을 멈출수 있다. 다른 검사는 조종점들의 린접한 쌍사이의 거리를 검사하는것이다. 또는 매 부분에서 조종점들의 모임이 거의 직선경로를 따를 때 부분분할을 멈출수 있다.

부분분할방법은 임의의 차수의 베지에곡선에 적용될수 있다. $n-1$차베지에다항식에 대하여 첫번째 부분분할걸음에서 곡선의 매 절반에 대한 $2n$개 조종점들은

$$\mathbf{p}_{1,k} = \frac{1}{2^k}\sum_{i=0}^{k} C(k,i)\mathbf{p}_i, \qquad k = 0,1,2,\cdots,n$$
$$\mathbf{p}_{2,k} = \frac{1}{2^{n-k}}\sum_{i=k}^{n} C(n-k,n-i)\mathbf{p}_i \qquad (10-94)$$

이다. 여기서 $C(k, i)$, $C(n-k, n-i)$는 2항결수들이다.

부분분할방법은 매듭벡토르에 값들을 추가하여 비균일B스플라인에 직접 적용할수 있다. 그러나 일반적으로 이 방법은 베지에부분분할만큼 효률적이지 못하다.

14절. 스위프표현

립체모형화프로그람들은 많은 구성기법들을 제공한다. 스위프(sweep)표현은 평행이동, 회전 또는 기타 대칭을 가지는 3차원물체를 만드는데 쓸모 있다. 이런 물체들은 2차원형태와 공간구역에서 그 형태를 움직이게 하는 스위프를 지적하여 표현할수 있다. 원 및 직4각형과 같은 2차원기초요소들의 모임은 스위프표현을 위한 차림표선택항으로 제공될수 있다. 2차원도형을 얻는 다른 방법은 닫긴 스플라인곡선구성과 립체물체의 자름면자르기를 포함한다.

그림 10-53은 평행이동스위프를 설명한다. 그림 10-53 ㄱ의 주기스플라인곡선은 물체의 자름면을 정의한다. 다음에 조종점 $\mathbf{p}_0$~$\mathbf{p}_3$을 자름면에 수직인 직선경로를 따라 일정한 거리만큼씩 움직여 평행이동스위프를 수행한다. 이 경로의 구간들에서 자름면의 형태를 재현하고 그림 10-53 ㄴ에 보여 준 선그물구조표현을 얻기 위하여 스위프의 방향으로 련결선들의 모임을 긋는다.

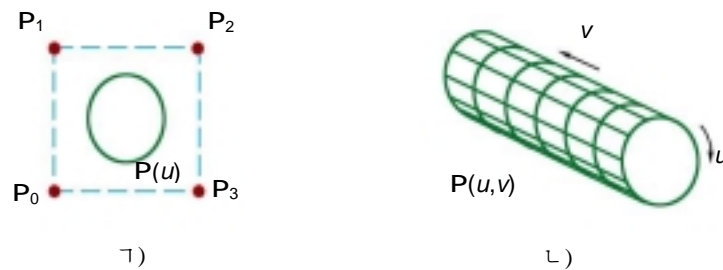



그림 10-53. 평행이동스위프에 의한 립체구성(ㄱ의 주기스플라인곡선의 조종점들을 평행이동시켜 ㄴ에 보여 준 립체를 만들며 그의 걸면은 점함수 $\mathbf{P}(u, v)$로 표현할수 있다.)

회전스위프를 리용하는 물체설계의 실례를 그림 10-54에 보여 주었다. 이때 주기스플라인자름면은 그림 10-54 ㄴ에 보여 준 선그물구조모형을 만들기 위하여 자름면에서 지적된 회전축주위로 회전한다. 임의의 축이 회전스위프에 대하여 선택될수 있다. 그림 10-54 ㄱ의 스플라인자름면에 수직인 회전축을 리용하면 2차원형태를 얻는다. 그러나 이 그림에 보여 준 자름면이 깊이를 가지면 다른것을 발생시키는 하나의 3차원물체를 가진다.

일반적으로 임의의 경로를 리용하는 스위프구성을 지적할수 있다. 회전스위프에 대하여 0~360°의 임의의 각범위에서 원경로를 따라 움직일수 있다. 비원경로를 표현하는 곡선함수와 경로에서의 이동거리를 지적할수 있다. 게다가 스위프경로를 따라 자름면의 형태 또는 크기를 변화시킬수 있다. 또는 형태를 공간구역을 따라 움직일 때 스위프경로에 따라 자름면의 방향을 변화시킬수 있다.

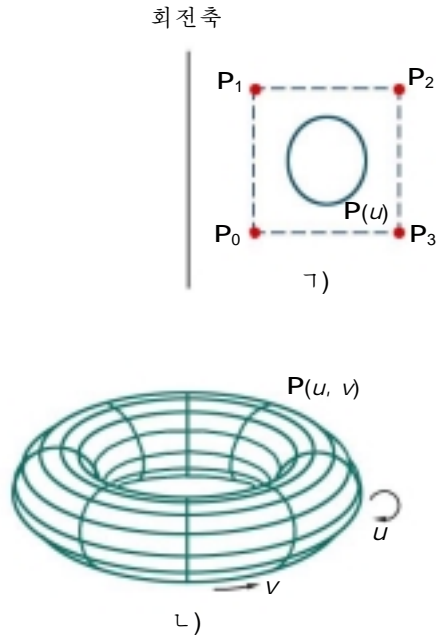



그림 10-54. 회전스위프에 의한 립체구성(ㄱ의 주기스플라인곡선의 조종점들을 주어 진 회전축주위로 회전시켜 ㄴ에 보여 준 립체를 얻는다. 그의 겉면은 점함수 **P**(*u*, *v*)로 표현할수 있다.)

15절. 립체구성기하방법

립체를 모형화하는 다른 기법은 겹치는 3차원물체들이 차지하는 체적들을 모임연산을 리용하여 결합하는것이다. **립체구성기하**(**CSG,** constructive solid geometry)라고 하는 이 모형화방법은 지적된 두 체적에 합, 사귐, 차연산을 적용하는 방법으로 새로운 체적을 만든다.

그림 10-55와 10-56에서는 모임연산을 리용하여 새로운 형태를 형성하는 실례를 보여 주었다. 그림 10-55 ㄱ에서는 블로크와 피라미드가 서로 붙어 있다. 합연산을 지적하면 그림 10-55 ㄴ에 보여 준 결합된 물체가 얻어 진다. 그림 10-56 ㄱ에서는 체적이 겹치는 블로크와 원기둥을 보여 주었다. 사귐연산을 리용하면 그림 10-56 ㄴ의 물체를 얻고 차연산을 진행하면 그림 10-56 ㄷ에 보여 준 립체를 얻는다.

CSG응용은 블로크, 피라미드, 원기둥, 원추, 구, 닫긴스플라인곡면과 같은 3차원물체(기초요소)들의 초기모임으로 시작한다. 기초요소들은 CSG프로그람에서 차림표의 선택항목으로 제공될수도 있고 기초요소자체를 스위프방법, 스플라인구성 기타 모형화절차들을 리용하여 만들수도 있다. CSG방

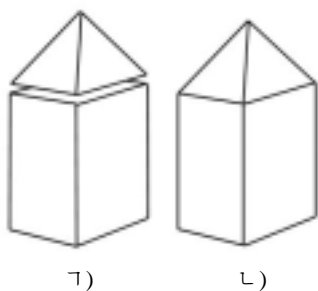



그림 10-55. ㄱ의 두 물체를 합연산에 의하여 결합하면 하나의 합성립체물체 ㄴ를 만든다.

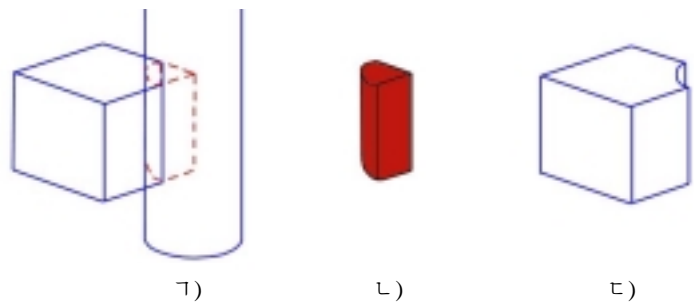



그림 10-56. ㄱ-겹치는 두 물체, ㄴ-사귐연산에 의하여 형성된 쐐기모양의 CSG물체, ㄷ-블로크의 체적에서 원기둥의 겹침체적을 덜어 내는 차연산으로 형성된 CSG물체

법을 리용하여 새로운 3차원형태를 만들기 위하여서는 먼저 2개의 기초요소를 선택하고 그것들을 공간의 어떤 위치에 끌어다 놓는다. 다음에 두개의 기초요소의 체적을 결합하는 연산(합, 사귐, 차)을 선택한다. 그러면 기초요소들외에 다른 물체를 형성하는데 리용할수 있는 새로운 물체를 가지게 된다. 마지막형태가 얻어 질 때까지 기초요소들과 매 걸음에서 만들어 진 물체들을 결합하여 새로운 형태만들기를 계속한다. 이 절차에 의하여 설계되는 물체는 2분나무로 표현된다. CSG물체에 대한 나무표현의 실례를 그림 10-57에 보여 주었다.

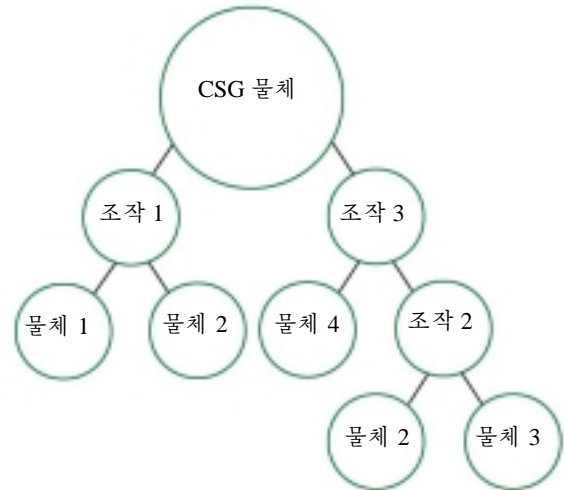



그림 10-57. 물체에 대한 CSG 나무표현

물체가 경계로 표현될 때에는 **광선투사**(ray-casting)방법이 립체구성기하조작을 실현하는데서 공통으로 리용되고 있다. 광선투사는 현시장치의 화소면에 대응하는 *xy*평면이 있는 세계자리표에서 합성물체를 구성하여 적용한다. 이 평면은 매개 화소위치로부터 결합될 물체를 지나가는 광선을 발사하기때문에 《발사면》이라고 한다(그림 10-58). 이때 매 광선경로를 따라서 걸면의 사귐점들을 결정하고 발사면으로부터의 거리에 따라 사귐점들을 정렬시킨다. 합성물체의 면경계는 지적되는 모임연산에 의하여 결정한다. CSG물체의 면경계를 광선투사에 의하여 결정하는 실례를 그림 10-59에 보여 주었다. 이 그림에서는 발사면에 수직인 화소광선의 경로와 두 기초요소의 *yz*자름면을 보여 주고 있다. 합연산으로 얻어 지는 새로운 체적은 두 기초요소가 차지하는 결합된 내부구역이다. 사귐연산에서의 새 체적은 두 기초요소의 공통내부구역이다. 그리고 차연산은 한 기초요소의 체적을 다른데서 덜어 낸다.

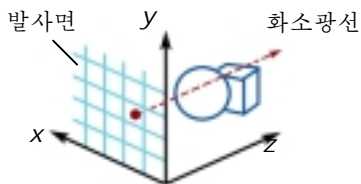
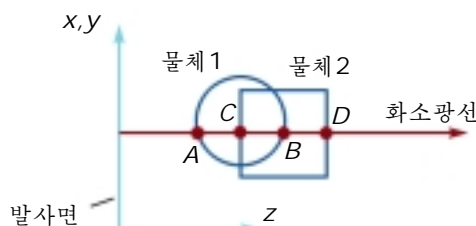



그림 10-58. 광선투사를 리용한 CSG 연산의 실현

매개 기초요소는 자체의 국부(모형화)자리표계에서 정의될수 있다. 그러면 합성된 형태는 두 기초요소를 세계자리표계의 겹침위치에 놓는 모형화변환행렬을 지적하여 만들수 있다. 이 모형화행렬의 역행렬은 그다음에 화소광선을 모형화자리표로 변환하는데 리용할수 있으며 거기서 개별적인 기초요소에 대하여 걸면사귐점을 계산한다. 그다음에 두 물체에 대한 걸면사귐점들을 정렬시키고 지적된 모임연산에 의하여 합성물체의 경계를 결정하는데 리용한다. 이 절차를 CSG나무에서 결합될 물체들의 매 쌍에 대하여 반복하여 하나하나의 물체를 만든다.

조 작	면의 경계
합	*A*, *D*
사 귐	*C*, *B*
차 (물체2-물체1)	*B*, *D*

ㄴ)

그림 10-59. 화소광선에 의한 걸면경계의 결정

일단 CSG물체가 만들어 지면 광선투사는 체적, 질량과 같은 물리적인 특성을 결정하는데 리용된다. 물체의 체적을 결정하기 위하여 그림 10-60에 보여 준바와 같이 발사면을 임의의 개수의 작은 바른4각형들로 나눈다. 다음에 위치 $(i,\ j)$의 바른4각형으로부터 광선경로를 따라서 면적 $A_{i\,j}$를 가지는 자름구간에 대한 물체의 체적 V_{ij}를

$$V_{ij} \approx A_{ij}\Delta z_{ij} \qquad (10-95)$$

와 같이 근사화할수 있다. 여기서 Δz_{ij}는 위치 $(i,\ j)$로부터의 광선에 따르는 물체의 깊이이다. 물체가 내부에 구멍을 가지고 있으면 Δz_{ij}는 광선이 지나가는 사귐점쌍들사이의 거리의 합이다. CSG물체의 총 체적은

$$V \approx \sum_{i,j} V_{ij} \qquad (10-96)$$

와 같이 계산된다.

물체의 밀도함수 $\rho(x, y, z)$가 주어 지면 위치 (i, j)로부터 광선을 따라 계산되는 질량은

$$m_{ij} \approx A_{ij}\int \rho(x_{ij}, y_{ij}, z)dz \qquad (10-97)$$

와 같이 근사화할수 있다. 여기서 1차원적분은 자주 적분을 실제로 계산하지 않고 밀도함수의 형에 따라 근사화할수 있다. CSG물체의 총 질량은

$$m \approx \sum_{i,j} m_{ij} \qquad (10-98)$$

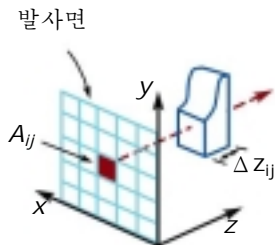



그림 10-60. 발사면의 미소면적 A_{ij}에 대한 광선경로를 따라서 물체의 체적을 결정한다.

와 같이 근사화된다. 질량중심, 관성모멘트와 같은 다른 물리적특성들도 류사한 계산에 의해 얻을수 있다. 발사면을 더 세분하면 물리적특성값들에 대한 근사계산을 개선할수 있다.

물체형태를 8분나무로 표현하면 8분공간의 공간적인 내용을 표현하는 나무구조를 주사하여 CSG절차의 모임연산을 실현할수 있다. 다음절에서 설명되는 이 절차에서는 결합될 두 물체가 차지하게 될 구역의 위치를 지정하기 위하여 단위바른6면체의 8분공간 및 부분8분공간들을 조사한다.

16절. 8분나무

일부 도형처리체계들에서는 립체물체를 표현하는데 **8분나무**(octree)라고 하는 계층적인 나무구조를 리용한다. 일반적으로 의학화상이나 물체의 자름면현시를 요구하는 응용에서는 8분나무표현을 리용한다. 이 나무구조는 매개 마디가 3차원공간의 구역에 대응하도록 구성된다. 립체에 대한 이 표현은 3차원물체의 공간적인 특성과 밀착되여 기억기를 줄이는 우점을 가지고 있다. 또한 물체내부에 대한 정보를 기억하는데 편리한 표현이다.

3차원공간에 대한 8분나무부호화절차는 **4분나무**(quadtree)부호화라고 하는 2차원공간에 대한 부호화방법의 확장이다. 4분나무는 2차원구역(보통 바른4각형)을 4분구로 련속적으로 나누는 방법으로 만든다. 4분나무의 매 마디는 구역의 매 4분구에 1개씩 4개의 자료원소를 가진다(그림 10-61). 어떤 4분구안의 모든 화소들이 동일한 색을 가지면(동질4분구) 대응하는 자료원소는 그 색으로 기억시킨다. 또 4분구가 동질이라는것을 지적하기 위하여 자료원소의 기발을 올린다. 그림 10-61의 4분구 2의 모든 화소들이 붉은색이라고 하자. 그러면 붉은색에 대한 색코드가 마디의 자료원소 2에 넣어 진다. 그렇지 않으면 4분구는 이질이라고 하며 그 4분구자체는 4분구들로 다시 나뉘여 진다(그림 10-

62). 이때 마디안의 대응하는 자료원소에는 4분구가 이질이라고 기발로 표시하며 4분나무의 다음마디에로의 지적자를 기억시킨다.

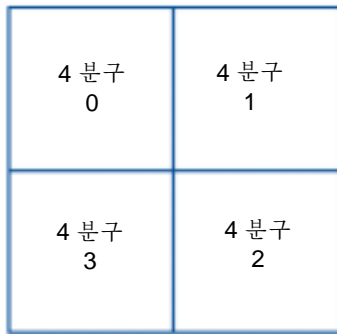
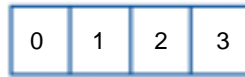



그림 10-61. 번호 매겨 진 4분구들로 나뉘여 지는 2차원공간의 구역과 4개의 자료원소를 가지는 관련4분나무마디

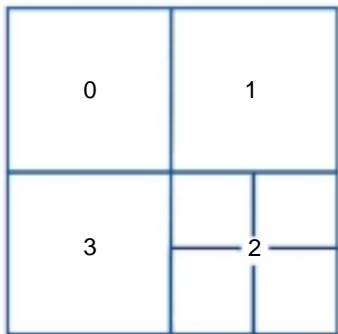



그림 10-62. 4 분구나누기의 2 개 준위를 가지는 2 차원공간의 구역과 관련 4 분나무표현

4분나무를 발생시키는 알고리듬은 화소세기값들을 검사하고 4분나무의 마디들을 해당하게 설정한다. 본래공간의 매 4분구가 단일한 색을 가지면 이 4분나무는 1개의 마디만을 가진다. 공간의 이질구역에 대하여서는 모든 4분구들이 동질로 될 때까지 4분구로의 세분을 계속한다. 그림 10-63에서는 구역의 다른 모든 부분들의 색과는 다른 색으로 된 부분을 포함하고 있는 구역에 대한 4분나무표현을 보여 주었다.

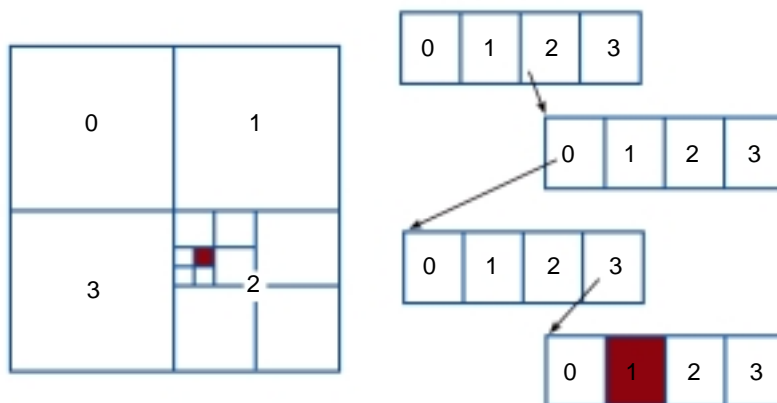



그림 10-63. 옹근배경에서 하나의 전경색화소를 포함하는 구역에 대한 4 분나무표현

4분나무부호화는 공간구역에 큰 색부분이 존재할 때 기억기를 대단히 절약할수 있다. 왜냐하면 매개 단일색부분이 하나의 마디로 표현되기때문이다. $2^n \times 2^n$화소를 포함하는 부분에 대한 4분나무표현은 최대 n개 준위를 가진다. 4분나무의 매개 마디는 최대 4개의 자식마디를 가진다.

8분나무부호화방법은 3차원공간구역(보통 바른6면체)을 8분공간들로 나누고 나무의 매 마디에 8개의 자료원소를 기억시킨다(그림 10-64). 3차원공간의 개별적인 요소들을 체적요소 또는 립체소라고 한다. 8분공간의 모든 립체소들이 동일한 형일 때 마디의 대응하는 자료원소에 이 형의 값을 기억시킨다. 공간의 빈 구역은 《void》립체소형으로 표현한다. 임의의 이질8분공간은 8분공간들로 부분분할하며 마디의 대응하는 자료원소는 8분나무의 다음마디를 가리키게 한다. 8분나무의 발생절차는 4분나무에서와 류사하다. 즉 매 8분공간의 립체소들을 검사하고 공간구역이 동질8분공간으로 될 때까지 8분공간부분분할을 계속한다. 8분나무의 매 마디는 0부터 8개까지의 자식마디를 가질수 있다.

8분나무발생알고리듬은 다각형그물, 곡면쪼각, 립체구성기하와 같은 임의의 형식의 물체정의를 받아 들이도록 구조화될수 있다. 물체의 최소 및 최대자리표값을 리용하여 물체주위에 직6면체를 정의할수 있다. 다음에 물체를 포함하는 3차원공간의 이 구역을 8분공간별로 검사하여 8분나무표현을 얻는다.

일단 립체물체에 대한 8분나무표현이 설정되면 여러가지 처리루틴들을 립체에 적용할수 있다. 공간의 동일한 구역에 대한 두 물체의 8분나무표현에 모임연산알고리듬을 적용한다. 합연산은 매 입력물체의 구역들의 결합으로 새로운 8분나무를 만든다. 이와 류사하게 사귐 및 차연산은 2개의 8분나무들에서 겹침구역을 찾는 방법으로 수행한다. 이때 새로운 8분나무는 두 물체가 겹치는 8분공간을 기억시키거나 다른것을 뽑아 낸 한 물체가 차지하는 구역만을 기억시켜 만든다.

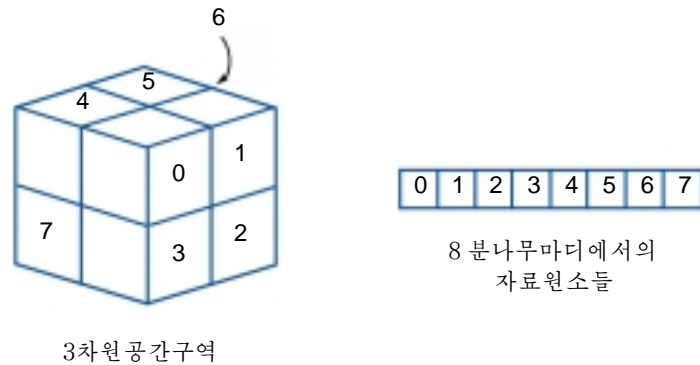



그림 10-64. 번호가 매겨 진 8 분공간으로 나뉘여 지는 3 차원공간 구역과 그에 해당한 8 개의 자료원소를 가지는 8 분나무마디

3차원8분나무에 대한 회전은 차지하는 8분공간들에 변환을 적용하는 방법으로 수행한다. 보이는 면식별은 8분공간들을 앞에서 뒤로 조사하는 방법으로 수행한다. 검출되는 첫번째 물체는 보이며 따라서 정보는 현시를 위한 4분나무표현에 전송되게 된다.

17절. BSP나무

이 표현방법은 매 걸음에서 공간을 8개가 아니라 두개의 부분으로 나눈다는것을 내놓으면 8분나무부호화와 류사하다. **2분공간분할**(BSP, binary space-partitioning)나무는 장면을 매 걸음에서 임의의 위치와 방향을 가질수 있는 평면을 리용하여 두 부분으로 부분분할한다. 8분나무부호화에서 장면은 매

걸음에서 직각자리표평면들에 평행인 서로 수직인 3개의 평면에 의해 부분분할된다.

공간을 적응적으로 부분분할하기 위하여 BSP나무에서는 물체의 공간적인 분포에 알맞게 자름면의 위치와 방향을 정할수 있기때문에 보다 효률적으로 분할한다. 이것은 8분나무에 비하여 장면에 대한 나무표현의 깊이를 줄일수 있으며 따라서 나무의 탐색시간을 줄인다. 이밖에 BSP나무는 보이는면식별과 광선추적알고리듬에서의 공간분할에 쓸모 있다.

18절. 프락탈기하방법

앞절들에서 고찰한 모든 물체표현들에서는 유클리드기하방법을 리용하였다. 즉 물체의 형태가 방정식으로 표현되였다. 이 방법들은 제작된 물체 즉 원활한 곡면과 규칙적인 형태를 가지는것들을 표현하는데는 충분하다. 그러나 산과 구름과 같은 자연의 물체들은 불규칙적인 또는 쪼각화된 특징을 가지며 유클리드방법은 이 물체들을 현실감 있게 모형화하지 못한다. 자연의 물체들은 **프락탈기하방법**(fractal-geometry method)에 의하여 현실감 있게 표현할수 있다. 여기서는 물체를 모형화하는데 방정식대신에 절차들을 리용한다. 예상할수 있는바와 같이 수속적으로 정의되는 물체들은 방정식으로 표현되는 물체와 전혀 다른 특성을 가진다. 물체에 대한 프락탈기하표현은 자연현상들의 특징을 표현하고 설명하는 많은 분야들에서 공통적으로 적용된다. 프락탈방법은 콤퓨터도형처리에서 자연물체들을 현시하며 여러가지 수학 및 물리학체계들을 가시화하는데 리용된다.

프락탈물체는 매개 점에서 끝 없는 세부를 가지며 물체의 전반적인 모습과 물체부분들사이에 어떤 자기상사성이 있다는 두가지 기본적인 특성을 가진다. 물체의 자기상사성은 프락탈의 표현방법에 따라 서로 다른 형식을 취할수 있다. 프락탈물체는 물체의 부분들에서의 세부를 만드는 반복조작을 지적하는 절차에 의해 표현된다. 자연의 물체들은 리론적으로 무한번 반복하는 절차에 의해 표현된다. 물론 자연물체의 도형처리현시는 유한번의 걸음에 의하여 만들어 진다.

련속적인 유클리드형태는 아무리 복잡할지라도 확대하면 나중에는 쭉 펴진 확대된 보임상이 얻어 진다. 그러나 프락탈물체를 확대하면 본래의 보임상에서와 같은 세부가 확대보임상에서 계속된다. 하늘을 배경으로 보이는 산은 그것을 점점 더 가까운 위치에서 볼 때에도 같은 터실터실한 모양을 계속 가진다(그림 10-65). 산으로 가까이 가면 개별적인 산세와 큰 돌의 더 작은 세부가 분명해 진다. 더 가까이 가면 바위의 륜곽을, 그다음은 돌, 또 그다음은 모래알을 보게 된다. 매 걸음에서 륜곽선은 더 많은 꼬임과 방향변화를 보여 준다. 모래알을 취하여 그것을 현미경으로 보면 분자준위에서 반복되는 동일한 세부를 다시 보게 될것이다. 해안선과 함께 식물과 구름의 변두리도 류사한 모양으로 표현된다.

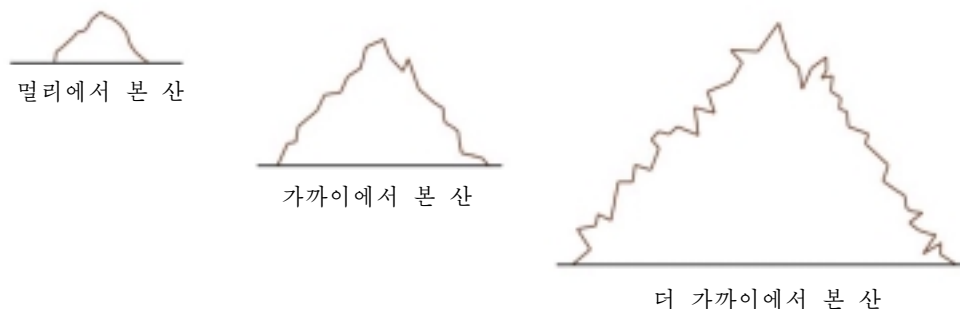



그림 10-65. 서로 다른 확대준위들에서 산의 륜곽선의 울퉁불퉁한 모양

도형처리에서 프락탈물체의 확대상은 더 작은 창문을 선택하고 새 창문에서 세부를 만들기 위하여 프락탈절차를 반복하는 방법으로 얻는다. 프락탈물체는 끝없는 세부를 가지기때문에 그것은 일정한 크기를 가지지 않는다. 점점 더 많은 세부를 고찰할 때 물체의 크기는 무한대에로 향하지만 그 물체의 자리표범위는 공간의 유한한 구역에 남아 있다.

물체세부의 변화정도는 프락탈차원이라고 부르는 수에 의하여 표현할수 있다. 유클리드차원과 달리 이 수는 꼭 옹근수로 되는것이 아니다. 물체의 프락탈차원은 때때로 분수차원이라고도 하는데 이것은 이름 《프락탈》(fractal)에 기초하고 있다.

프락탈방법들은 다양한 자연현상들을 모형화하는데 편리하다는것이 판명되였다. 도형처리응용에서 프락탈표현은 지형, 구름, 물, 나무와 기타 식물들, 깃털, 털가죽, 여러가지 결문양을 모형화하며 아름다운 무늬를 만드는데 리용된다. 다른 분야에서 프락탈무늬는 별의 분포, 강의 분포, 달의 분화구의 분포, 비내림마당, 주식시장변화, 음악, 교통흐름, 도시의 재산리용, 수값해석할 때 수렴구역의 경계에서 찾아 보게 된다.

프락탈발생절차

프락탈물체는 지적된 변환함수를 공간구역안의 점들에 반복적용하여 만든다. $\mathbf{P}_0=(x_0, y_0, z_0)$이 선택된 초기점이라면 변환함수 F의 반복은

$$\mathbf{P}_1 = F(\mathbf{P}_0), \quad \mathbf{P}_2 = F(\mathbf{P}_1), \quad \mathbf{P}_3 = F(\mathbf{P}_2), \quad \cdots \qquad (10-99)$$

의 계산에 의해 련속적인 세부준위들을 만든다.

일반적으로 변환함수는 지적된 점모임에 적용할수도 있고 직선, 곡선, 색구역, 면, 립체물체와 같은 기초요소들의 초기모임에 적용할수도 있다. 또한 매 반복에서 확정적인 또는 우연적인 발생절차를 리용할수 있다. 변환함수는 기하학적인 변환(비례변환, 평행이동, 회전)에 의하여 정의될수도 있고 비선형자리표변환과 결심파라메터에 의하여 설정될수도 있다.

정의에 의하여 프락탈물체는 비록 끝 없는 세부를 가지지만 변환함수는 유한번 적용한다. 따라서 현시하는 물체는 실제로는 유한차원을 가진다. 수속적인 표현은 변환의 수를 증가시켜 더욱더 많은 세부를 만들면 《진짜》프락탈에 접근하게 된다. 마지막에 물체를 현시할 때 물체의 세부의 량은 수행된 반복회수와 현시체계의 해상도에 관계된다. 화소의 크기보다 더 작은 세부변화는 현시할수 없다. 더 많은 물체세부를 보기 위하여서는 선택된 부분을 확대하고 변환을 반복한다.

프락탈의 분류

자기상사(self-similar)프락탈은 전체 물체의 축소판들을 가진다. 물체의 부분들은 초기형태에서 시작하여 비례결수 s를 전체 형태에 적용하는 방법으로 만든다. 모든 부분들에 대하여 동일한 비례결수 s를 리용할수도 있고 물체의 서로 다른 축소판들에 대하여 서로 다른 비례결수를 리용할수도 있다. 또한 축소판들에 우연적인 변화를 적용하면 그 프락탈을 통계적자기상사라고 한다. 이때 부분들은 동일한 통계적특성을 가진다. 통계적자기상사프락탈은 키나무, 떨기나무, 기타 식물들을 모형화하는데 공통적으로 리용된다.

자기아핀(self-affine)프락탈은 서로 다른 자리표방향에서 서로 다른 비례결수 s_x, s_y, s_z에 의하여 형성되는 부분들을 가진다. 그리고 통계적자기아핀프락탈을 얻기 위하여 우연변화를 포함시킬수 있다. 지형, 물, 구름들은 일반적으로 통계적자기아핀프락탈구성방법에 의하여 모형화된다.

불변프락탈모임(invariant fractal set)은 비선형변환에 의하여 형성된다. 이 부류의 프락탈에는 만델브로트모임과 같은 자기두제곱프락탈이 있는데 그것은 복소공간에서의 두제곱함수에 의해 형성된다.

그리고 역절차에 의하여 형성되는 프락탈을 자기역프락탈이라고 한다.

프락탈차원

프락탈물체의 세부변화는 **프락탈차원**(fractal dimension)이라고 부르는 수 D에 의하여 표현할수 있다. 이것은 물체의 울퉁불퉁함 또는 분렬의 척도이다. 더 터실터실하게 보이는 물체는 더 큰 프락탈차원을 가진다. 주어 진 프락탈차원 D값을 리용하여 프락탈물체를 만드는 몇가지 반복절차들을 설정할수 있다. 일반적으로 프락탈차원은 계산하기 힘들지만 구성된 물체의 특성으로부터 다른 절차들에 의하여서도 프락탈차원을 결정할수 있다.

하나의 비례결수 s에 의하여 만들어 지는 자기상사프락탈의 프락탈차원에 대한 식은 유클리드물체의 부분분할과 류사하게 얻어 진다. 그림 10-66에서는 단위선분, 바른4각형, 바른6면체에서 비례결수 s와 분할부분들의 개수 n사이의 관계를 보여 주었다. $s=1/2$일 때 단위선분(그림 10-66 ㄱ)은 두개의 같은 길이의 부분들로 나뉘여 진다. 류사하게 그림 10-66 ㄴ의 바른4각형은 4개의 같은 면적의 부분들로 나뉘여 지며 바른6면체(그림 10-66 ㄷ)는 8개의 같은 체적의 부분들로 나뉘여 진다. 이 매 물체들에서 분할부분들의 개수와 비례결수사이의 관계는 $n \cdot s^{D_E} = 1$이다. 유클리드물체와 류사하게 자기상사물체에 대한 프락탈차원 D는

$$ns^D = 1 \qquad (10-100)$$

로부터 얻을수 있다. 이 식을 D에 대하여 풀면 프락탈차원은

$$D = \frac{\ln n}{\ln(1/s)} \qquad (10-101)$$

이다. 매 부분들에 대하여 서로 다른 비례결수를 가지고 만든 자기상사프락탈에 대한 프락탈차원은 음함수적인 관계

$$\sum_{k=1}^{n} s_k^D = 1 \qquad (10-102)$$

로부터 얻는다. 여기서 s_k는 분할부분의 번호 k에 대한 비례결수이다.

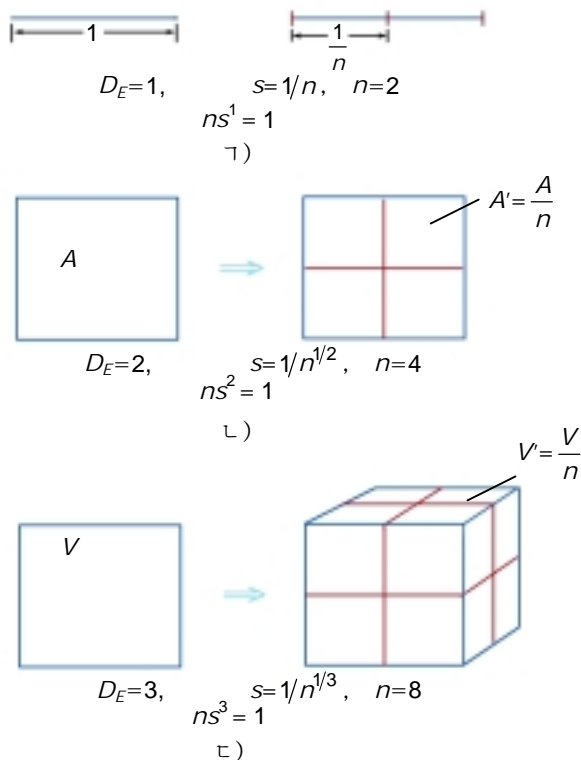



그림 10-66. 비례결수 $s=1/2$ 을 리용하여 유클리드차원이 $D_E=1$(ㄱ), $D_E=2$(ㄴ), $D_E=3$(ㄷ) 인 물체를 부분분할

그림 10-66에서는 간단한 형태(직선, 직4각형, 직6면체)들의 부분분할을 고찰하였다. 곡선과 곡면을 가지는 물체를 비롯하여 더 복잡한 형태에서는 분할부분의 구조와 특성을 결정하는것이 더 힘들어 진다. 일반적인 형태의 물체에서는 물체의 분할부분들을 간단한 형태로 근사화하는 위상기하학적인 덮기방법(topological covering method)을 리용할수 있다. 실례로 부분분할된 곡선은 선분들로 근사화할수 있으며 부분분할된 다각형은 작은 바른4각형 또는 직4각형으로 근사화할수 있다. 원, 구, 원기둥과 같은 다른 덮기형태들도 부분분할된 물체의 특징을 근사화하는데 리용될수 있다. 일반적으로 수학에서 덮기방법은 작은 덮기물체들의 특성을 합하는 방법으로 물체의 길이, 면적, 체적과 같은 기하학적특성을 결정하는데 리용되고 있다. 또한 덮기방법은 일부 물체들의 프락탈차원을 결정하는데도 리용할수 있다.

위상기하학적인 덮기개념은 본래 기하학적인 특징의 의미를 비표준형태들에 확장하는데 리용되였다. 원 또는 구를 리용하는 덮기방법의 확장으로 하우스도르프-베씨꼬위츠(Hausdorff-Besicovitch)차원 또는 분수차원(fractional dimension)이 나왔다. 하우스도르프-베씨꼬위츠차원은 일부 물체들에서 프락탈차원으로 리용할수도 있지만 일반적으로 평가하기가 힘들다. 물체의 보다 일반적인 프락탈차원은 직4각형 또는 직6면체를 리용하는 칸덮기방법으로 평가한다. 그림 10-67에서는 칸덮기의 일반개념을 보여 주었다. 여기서는 불규칙적인 큰 경계안의 구역을 작은 덮기직4각형구역들의 합으로 근사화할수 있다.

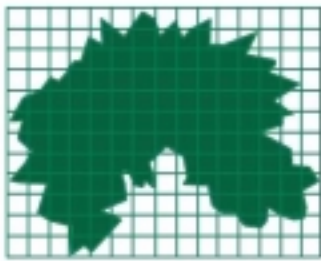



그림 10-67. 불규칙적인 모양의 물체를 직4각형으로 덮기

칸덮기는 먼저 물체의 자리표범위를 결정한 다음 주어 진 비례결수를 리용하여 물체를 작은 칸들로 부분분할하는 방법으로 진행한다. 물체를 덮기 위하여 취하는 칸의 수 n을 칸차원이라고 하는데 n은 물체의 프락탈차원 D에 관계된다. 단일한 비례결수 s를 가지는 통계적으로 자기상사인 물체들에 대하여서는 물체를 바른4각형이나 바른6면체로 덮을수 있다. 다음에 칸의 개수 n을 계산하고 프락탈차원을 평가하기 위하여 식 10-101을 리용한다. 자기아핀물체들에서는 각 방향이 서로 다르게 비례변환되기때문에 물체를 직4각형칸으로 덮는다. 이 경우 칸의 개수 n을 프락탈차원을 결정하기 위하여 아핀변환파라메터와 함께 리용한다.

물체의 프락탈차원은 단순히 물체를 표시하는데 필요한 파라메터들의 최소수인 대응하는 유클리드차원(위상기하학적인 차원)보다 항상 크다. 유클리드곡선은 1차원이며 유클리드면은 2차원, 유클리드립체는 3차원이다.

2차원평면에 완전히 놓이는 프락탈곡선의 프락탈차원 D는 1(곡선의 유클리드차원)보다 크다. D가 1에 가까울수록 프락탈곡선은 더 원활하다. $D=2$이면 페아노(Peano)곡선 즉 2차원공간의 유한구역을 완전히 채우는 《곡선》이다. $2<D<3$이면 곡선은 자체로 사귀며 구역을 무한번 덮을수 있다. 프락탈곡선은 해안선과 같은 자연물체의 경계를 모형화하는데 리용될수 있다.

공간적인 프락탈곡선(한 평면에 완전히 놓이지 않는)도 역시 1보다 큰 프락탈차원 D를 가지지만 D는 자체사귐이 없이 2보다도 클수 있다. 공간의 체적을 채우는 곡선은 차원 $D=3$을 가지며 자체사귐공간곡선은 프락탈차원 $D>3$을 가진다.

프락탈면은 일반적으로 $2<D\leq3$범위내의 차원을 가진다. $D=3$이면 면은 공간의 체적을 채운다. 그리고 $D>3$이면 체적들의 겹침이 있다. 지형과 구름, 물은 일반적으로 프락탈면에 의해 모형화된다.

프락탈립체의 차원은 보통 $3<D\leq4$범위에 있다. 또한 $D>4$이면 자체겹침물체가 얻어 진다. 실례로 프락탈립체는 공간구역내의 수증기밀도, 온도와 같은 특성을 모형화하는데 리용될수 있다.

확정적자기상사프락탈의 기하학적인 구성

확정적(비우연)자기상사프락탈을 기하학적으로 만들기 위하여서는 초기형태(initiator)라고 부르는 주어 진 기하학적형태로부터 시작한다. 초기형태의 분할부분들은 다음에 발생무늬(generator)라고 부르는 무늬에 의하여 교체된다.

실례로 그림 10-68에 보여 준 초기형태와 발생무늬를 리용하면 그림 10-69에 보여 준 눈송이무늬 즉 코흐(Koch)곡선을 만들수 있다. 초기형태의 매개 선분은 매 걸음에서 4개의 같은 길이선분으로 교체된다. 비례결수는 1/3이며 따라서 프락탈차원은 D=ln4/ln3≈1.2619이다. 또한 초기형태의 매개 선분의 길이는 매 걸음에서 4/3배로 증가하며 프락탈곡선의 길이는 곡선에 더 많은 세부가 추가됨에 따라 무한대로 되게 된다(그림 10-70). 자기상사프락탈곡선의 다른 구성실례를 그림 10-71에서 보여주었다. 이 실례들은 프락탈차원이 더 높을수록 물체가 더 터실터실하게 된다는것을 보여 준다.

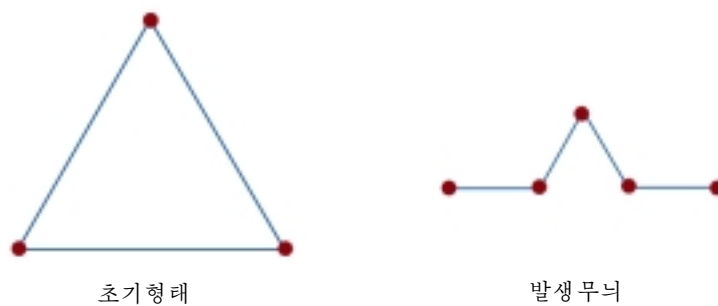



그림 10-68. 코흐곡선에 대한 초기형태와 발생무늬

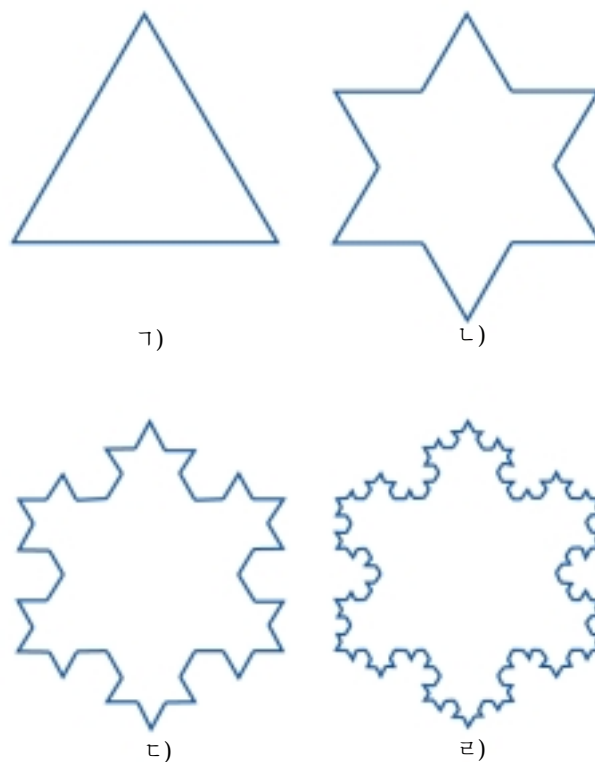



그림 10-69. 코흐곡선의 발생에서 첫 3개의 반복

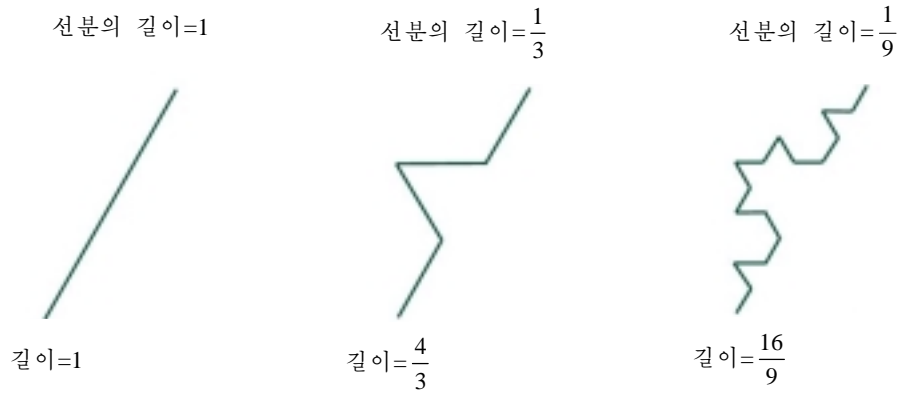



그림 10-70. 코흐곡선의 매변의 길이는 매 걸음에서 4/3 배로 길어 지며 한편 선분의 길이는 1/3 로 짧아 진다.

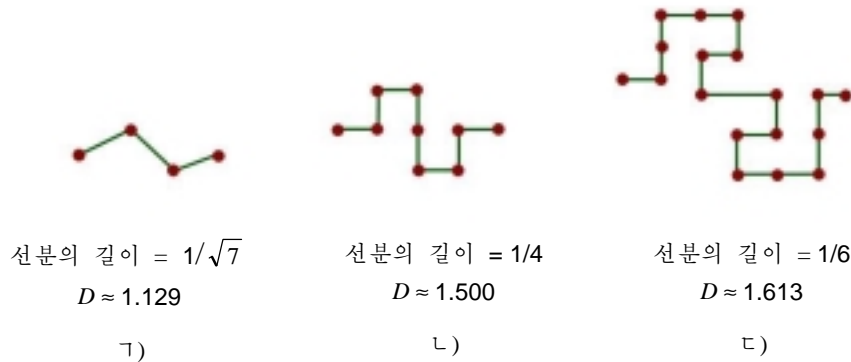



그림 10-71. 자기상사곡선의 구성과 프락탈차원

또한 다중분리요소들을 가지는 발생무늬를 리용할수도 있다. 복합발생무늬의 몇가지 실례를 그림 10-72에서 보여 주었다. 복합발생무늬를 가지는 우연적인 변화를 리용하면 해안선을 따르는 섬의 분포와 같은 복합부분들을 가지는 여러가지 자연물체들을 모형화할수 있다.

그림 10-73에서는 다중비례결수를 리용한 자기상사구성의 실례를 보여 주었다. 이 물체의 프락탈차원은 식 10-102로부터 결정된다.

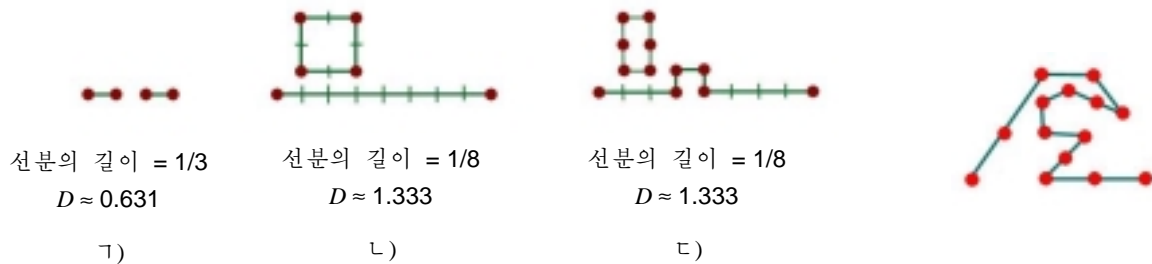



그림 10-72. 다중분리부분들을 가지는 발생무늬

그림 10-73. 눈송이 채우기페아노곡선

면에 대한 자기상사프락탈의 구성실례로서 그림 10-74에 보여 준 바른4면체를 결수 1/2로 비례

변환하고 그다음 비례변환된 물체를 4면체의 본래의 매 4개 면에 놓는다. 그러면 본래의 4면체의 매개 면은 6개의 작은 면들로 변환되며 본래 면의 면적은 3/2배로 된다. 이 면의 프락탈차원은

$$D = \frac{\ln 6}{\ln 2} \approx 2.58496$$

이다. 이것은 상당한 쪼각면을 가리킨다.

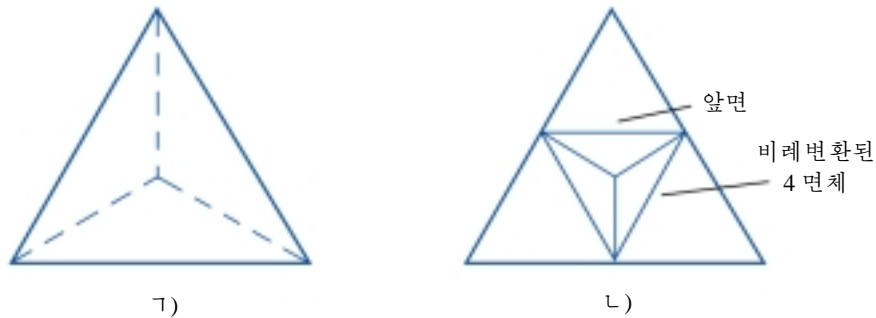



그림 10-74. ㄱ의 4면체를 결수 1/2에 의하여 비례변환하고 비례변환된것을 본래 4면체의 한면에 위치 정하면 프락탈면 ㄴ를 만든다.

자기상사프락탈물체를 만드는 다른 방법은 더 많은 면구역을 추가하는것이 아니라 주어 진 초기형태에 구멍을 내는것이다. 그림 10-75에서는 이 방법으로 만든 프락탈물체의 몇가지 실례를 보여 주었다.

그림 10-75. 초기형태에서 분할부분들을 덜어 내는 발생무늬로 만든 자기상사 3차원프락탈

통계적자기상사프락탈의 기하학적인 구성

자기상사프락탈의 기하학적구성에 일부 우연성을 도입할수 있는 한가지 방법은 매 걸음에서 발생무늬를 미리 정의된 형태들의 모임으로부터 우연적으로 선택하는것이다. 우연적인 자기상사물체를 발생시키는 다른 방법은 자리표변위를 우연적으로 계산하는것이다. 실례로 그림 10-76에서는 매 걸음에서 우연적인 중점변위거리를 선택하여 우연적인 눈송이무늬를 만들고 있다.

그림 10-76. 우연적인 중점변위를 리용하는 수정된 《눈송이》무늬

나무와 기타 식물들은 이러한 기하학적방법에 의하여 현시할수 있다. 그림 10-77에서는 고사리에 대한 자기상사구성을 보여 주었다. 이 그림의 ㄱ에서 매개 가지는 전체 물체의 축소판이며 ㄴ에서는 매 가지에 꼬임이 적용되고 완전히 실감처리된 고사리를 보여 주었다. 이 방법의 다른 실례를 그림 10-78에서 보여 주었다. 여기서는 우연적인 비례결수와 가지치기방향을 잎의 결무늬를 모형화하는데 리용하고 있다.

ㄱ) ㄴ)

그림 10-77. 고사리에 대한 자기상사구성

그림 10-78. 락엽에서 결의 우연적인 자기상사구성(잎사귀의 경계는 결의 생장범위이다.)

일단 프락탈물체들의 모임이 만들어 지면 여러가지로 변환된 프락탈물체들의 구체레들을 가지고 장면을 모형화할수 있다. 그림 10-79에서는 간단한 프락탈나무의 구체레들을 보여 주었다. 그림 10-80에는 프락탈숲이 현시되여 있다.

그림 10-79. 물체의 여러가지 구체레를 리용한 장면의 모형화(프락탈잎사귀들을 나무에 붙이고 나무의 여러가지 구체레들을 작은 숲을 형성하는데 리용하고 있다. 풀은 풀색원추의 여러가지 구체레로 모형화하고 있다.)

비례변환뿐아니라 꼬임함수를 우연적자기상사의 가지를 만드는데 적용하면 일부 나무들의 옹이투성이와 비틀어 진 형태를 모형화할수 있다. 이 방법을 그림 10-81에서 보여 주었다. 이 그림의 왼쪽에 있는 끝으로 가면서 가늘어 지는 원기둥에 변환을 적용하여 라선, 타래선, 우연꼬임무늬(왼쪽에서 오른쪽으로)를 만들수 있다. 우연꼬임으로 모형화한 나무를 그림 10-82에서 보여 주었다. 이 현시에서 나무껍질은 뒤에서 설명하는바와 같이 곰보만들기와 곰보무늬에서의 프락탈브라운변화를 리용하여 모형화한다.

그림 10-80. 잎사귀, 소나무잎, 풀, 나무껍질의 다중구체레에 의하여 만든 프락탈숲

그림 10-81. 라선, 타래선, 우연꼬임에 의한 나무가지의 모형화

그림 10-82. 우연적으로 비틀어 지는 곡선에 의해 모형화된 나무가지

아핀프락탈구성방법

물체의 특징을 프락탈적인 브라운운동으로 모형화하는 아핀프락탈방법을 리용하면 지형과 다른 자연물체들에 대한 대단히 현실감 있는 표현을 얻을수 있다. 이것은 표준브라운운동의 확장으로서 기체나 다른 류체립자들의 산만한 갈지자모양의 이동을 표현하는 《우연적인 움직임》의 형식이다. 그림 10-83에서는 평면에서의 우연적움직임의 경로를 보여 주었다. 주어 진 위치에서 시작하여 우연적인 방향으로 우연적인 길이의 선분들을 발생시킨다. 다음 첫번째 선분의 끝점에 가서 처리를 반복한다. 이 절차를 임의

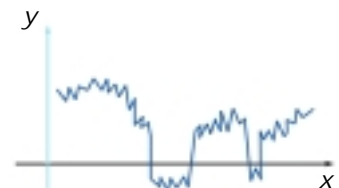



그림 10-83. 평면에서 브라운운동(우연움직임)의 실례

의 개수의 선분에 대하여 반복하는데 선경로의 통계적특성은 임의의 시간구간 t에서 계산할수 있다. 프락탈적인 브라운운동은 브라운운동을 표현하는 통계적분포에 보충적인 파라메터를 추가하여 얻는다. 이 보충적인 파라메터는 《운동》경로에 대한 프락탈차원을 설정한다.

단일한 프락탈적인 브라운경로는 프락탈곡선을 모형화하는데 리용할수 있다. 지면의 격자우에서 우연적인 프락탈브라운높이들의 2차원배렬을 만들어 놓으면 높이들을 련결한 다각형곡면쪼각들의 모임을 형성하여 산의 걸면을 모형화할수 있다. 만약 우연높이를 구면에서 발생시키면 행성의 산, 골짜기, 대양을 모형화할수 있다. 그림 10-84에서는 브라운운동이 행성걸면에서 높이의 변화를 만드는데 리용되였다. 여기서는 제일 낮은 높이가 푸른색(대양)으로 그려 지고 제일 높은 높이가 흰색(산의 눈)을 가지도록 색부호화하였다. 프락탈적인 브라운운동은 전경에 지형특징을 만드는데도 리용되였다. 분화구는 강줄기, 비무늬, 다른 류사한 물체계들의 분포를 세세히 표현하는 아핀프락탈절차를 리용하여 우연적인 직경을 가지고 우연적인 위치에 만들었다.

그림 10-84. 전경에 분화구를 추가한 프락탈브라운운동행성의 걸면에서 본 브라운운동행성

프락탈적인 브라운운동의 계산에서 프락탈차원을 조정하면 지형의 울퉁불퉁함을 변화시킬수 있다. $D \approx 2.15$근방의 프락탈차원값은 현실감 있는 산의 모습을 만든다. 한편 3.0에 가까운 더 큰 값들은 류별나게 보이는 지구밖의 풍경을 만드는데 리용할수 있다. 또한 골짜기를 깊게 하고 산봉우리를 높게 하기 위하여 계산된 높이들을 비례변환할수도 있다. 프락탈절차에 의하여 모형화할수 있는 지형모습들의 일부 실례를 그림 10-85에서 보여 주었다. 프락탈산우의 프락탈구름으로 모형화한 장면을 그림 10-86에서 보여 주었다.

우연중점변위법

프락탈적인 브라운운동계산에는 시간이 든다. 왜냐하면 지면우의 지형의 높이자리표는 코시누스 및 시누스항들의 합으로 이루어 진 푸리에합렬로 계산되기때문이다. 일반적으로 고속푸리에변환(FFT)방법들을 리용하고 있지만 프락탈산의 장면을 만드는데는 여전히 느리다. 그러므로 지형 및 기타 자연현상들에 대한 프락탈적인 브라운운동의 표현을 근사화하기 위하여 기하학적구성에서 리용되는 우연변위방법과 류사한 빠른 **우연중점변위법**(random midpoint-displacement method)이 개발되였다. 이 방법은 본래 류별난 모양의 지형과 행성이 나오는 과학환상영화에서 동화상프레임들을 만드는데 리용되였다. 중점변위법은 텔레비죤광고를 위한 동화를 비롯하여 많은 응용들에서 일반적으로 리용되고 있다.

ㄱ) ㄴ) ㄷ)

그림 10-85. 프락탈적인 브라운운동에 의하여 모형화된 지형특징들에서의 변화

그림 10-86. 프락탈구름과 산으로 모형화한 장면

비록 우연중점변위법은 프락탈적인 브라운운동에서보다 계산이 빠르지만 지형모습의 현실감에서는 좀 못하다. 그림 10-87에서는 xy평면에서 우연움직임의 경로발생을 위한 중점변위법을 보여 주었다. 선분에서 시작하여 선분의 중간위치에서 y값의 변위를 끝점들의 y값의 평균에 우연적편위를 더하여 계산한다.

$$y_{\text{mid}} = \frac{1}{2}[y(a) + y(b)] + r \tag{10-103}$$

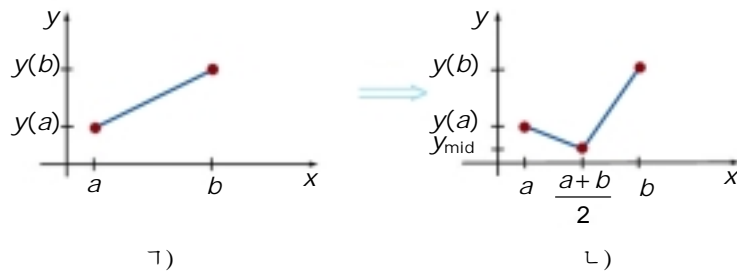



그림 10-87. 선분의 우연중점변위

프락탈적인 브라운운동을 근사화하기 위하여 r의 값은 평균이 0이고 분산이 $|b-a|^{2H}$에 비례하는 가우스분포로 선택한다. 여기서 H=2–D이며 D>1은 프락탈차원이다. 우연적편위를 얻는 다른 방법은 $r=sr_g|b-a|$를 취하는것이다. 여기서 파라메터 s는 선택된 《걸면의 울퉁불퉁함》을 표현하는 인자이고 r_g는 평균이 0이고 분산이 1인 가우스우연값이다. 가우스분포값을 얻는데는 표검색법을 리용할수 있다. 처리는 부분분할된 선분의 매 절반의 중간위치에서 변위된 y값을 계산하는 방법으로 반복한다. 그리고 부분분할은 분할된 선분이 어떤 미리 설정된 값보다 작아 질 때까지 계속한다. 매 걸음에서 우연변수 r의 값은 감소된다. 왜냐하면 그것은 부분분할된 선분의 너비 $|b-a|$에 비례하기때문이다. 그림 10-88에서는 이 방법에 의하여 얻어 지는 프락탈곡선을 보여 주었다.

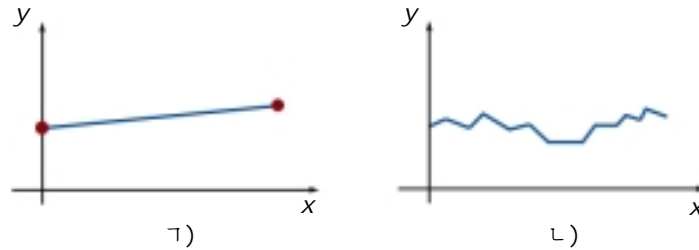



그림 10-88. 선분에서 우연중점변위절차를 4번 반복하여 만든 우연움직임경로

지형의 모양은 우연중점변위절차를 직4각형평면에 적용하여 만든다(그림 10-89). 지면의 4개의 매 구석(그림 10-89에서 a, b, c, d)에 높이 z값들을 할당하는것으로부터 시작한다. 그다음 지면을 매변의 중점에서 분할하여 5개의 새로운 격자위치 e, f, g, h, m을 얻는다. 중간위치 e, f, g, h에서의 높이는 제일 가까운 두 정점의 평균높이에 우연적인 편위를 더하여 계산할수 있다. 실례로 중간위치 e에서의 높이 z_e는 정점 a와 b를 리용하여 계산하며 중간위치 f에서의 높이는 정점 b와 c를 리용하여 계산한다.

$$z_e = (z_a + z_b)/2 + r_e, \qquad z_f = (z_b + z_c)/2 + r_f$$

우연값 r_e와 r_f는 평균이 0이고 분산이 격자간격의 $2H$제곱에 비례하는 가우스분포로부터 얻을수 있다. 여기서 H=3–D이며 D>2는 면의 프락탈차원이다. 그리고 우연적인 편위는 면의 울퉁불퉁함인자에 격자간격을 곱하고 평균이 0, 분산이 1인 가우스값에 대한 표검색값을 곱한 적으로 계산할수 있다. 지면의 중간위치 m에서의 높이 z_m은 위치 e와 g 또는 위치 f와 h를 리용하여 계산할수 있다. z_m은 평면의 4개 구석에 할당된 높이를 리용하여 계산할수도 있다.

$$z_m = (z_a + z_b + z_c + z_d)/4 + r_m$$

이런 처리를 격자간격이 선택된 값보다 더 작게 될 때까지 매 걸음에서 4개의 매 새로운 부분격자에 대하여 반복적용한다.

높이를 발생시킬 때 3각형면쪼각이 형성될수 있다. 그림 10-90에서는 첫번째 부분분할걸음에서 형성되는 8개의 면쪼각을 보여 주었다. 재귀의 매 준위에서 3각형은 더 작은 평면쪼각들로 련속적으로 부분분할된다. 부분분할처리가 끝나면 광원의 위치, 다른 조명파라메터들의 값, 지형에 대하여 선택한 색과 결문양을 가지고 곡면쪼각들을 실감처리한다.

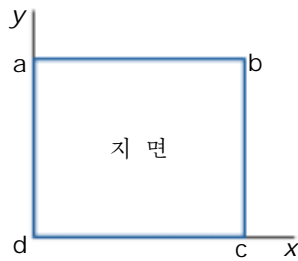



그림 10-89. 지형의 높이를 계산하는 우연중점변위절차의 첫 걸음에서 직4각형지면 ㄱ는 4개의 같은 간격의 부분격자 ㄴ로 부분분할된다.

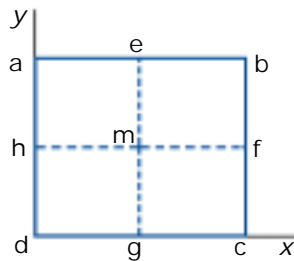



그림 10-90. 지형의 모양을 만들기 위한 우연중점변위절차의 첫번째 걸음에서 지면우에 형성되는 8개의 면쪼각

우연중점변위법은 지형외에 장면의 다른 요소들을 만드는데 적용될수 있다. 실례로 물결면의 모양 또는 지면우의 구름무늬를 얻는데 같은 방법을 리용할수 있다.

지형조종

중점변위법에 의하여 모형화되는 프락탈지형장면에서 봉우리와 골짜기의 배치를 조종하는 한가지 방법은 계산된 높이들을 지면의 서로 다른 구역들에서 어떤 구간안에 떨어 지도록 제한하는것이다. 이를 위하여 그림 10-91에 보여 준바와 같이 지면우에 조종면들을 설정한다. 그러면 지면의 매 중간점격자위치에서 조종면높이와 그 위치에서 계산되는 평균높이사이의 차에 관계되는 우연높이를 계산한다. 이 절차는 계산높이를 조종면높이의 미리 설정한 구간안에 들어 가도록 제한한다.

그림 10-91. 지면우의 조종면

조종면은 현존하는 산이나 기타 다른 구역의 지형상 특징을 매개 구역에 대한 등고선도의 높이를 리용하여 평면쪼각을 만드는 방법으로 모형화하는데 리용할수 있다. 또는 조종다각형의 정점들의 높이를 사용자들이 지형의 모양을 설계하기 위하여 설정할수도 있다. 조종면은 또한 임의의 형태를

가질수 있다. 다루기가 제일 쉬운것은 평면이지만 구면 또는 다른 곡면형태들도 리용할수 있다.

격자높이의 계산에는 우연중점변위법을 리용하지만 우연값들은 평균 μ와 표준편차 σ가 조종높이의 함수인 가우스분포로부터 선택한다. μ와 σ값들을 설정하는 한가지 방법은 그것들이 다같이 매 격자위치에서 계산된 평균높이와 미리 정의된 조종높이사이의 차에 비례하도록 하는것이다. 실례로 그림 10-89의 격자위치 e에 대하여서는 평균과 표준편차를

$$\mu_e = zc_e - (z_a + z_b)/2, \qquad \sigma_e = s|\mu_e|$$

와 같이 설정한다. 여기서 zc_e는 지면위치 e에서의 조종높이이고 $0<s<1$은 미리 설정한 비례결수이다. s값이 작으면(즉 $s<0.1$) 지형에서 굴곡이 심하지 않게 되며 s값이 크면 지형의 높이가 더 크게 오르내리게 된다.

평면형태의 조종면의 조종높이값을 결정하기 위하여서는 먼저 평면의 파라메터 A, B, C, D를 계산한다. 그러면 지면의 임의의 위치(x, y)에서 그 조종다각형을 포함하는 평면의 높이는

$$zc = (-Ax - By - D)/C$$

와 같이 계산된다. 이때 지면격자위치들에서의 조종높이계산에 증분방법을 리용할수 있다. 이 계산을 효률적으로 수행하기 위하여 먼저 그림 10-92에 보여 준바와 같이 지면을 xy위치들의 그물로 부분분할한다. 다음에 매개 조종다각형면을 지면에 투영한다. 그러면 주사선구역채우기와 류사한 절차를 리용하여 어느 위치들이 조종다각형의 투영안에 있는가를 결정할수 있다. 즉 다각형의 변들을 지나가는 지면그물의 매 y《주사선》에 대하여 주사선사귐점을 계산하고 어느 격자위치들이 조종다각형투영의 내부에 있는가를 결정한다. 그 격자위치들에서 조종높이는

$$zc_{i+1,j} = zc_{i,j} - \Delta x(A/C), \qquad zc_{i,j+1} = zc_{i,j} - \Delta y(B/C) \qquad (10-104)$$

와 같이 증분적으로 계산할수 있다. 여기서 Δx와 Δy는 x, y방향에서의 격자간격이다. 이 절차는 조종면격자위치들을 처리하는데 평행벡토르방법을 적용할 때 빠르다.

그림 10-93에서는 지면우의 지형, 물, 구름면에 조종면을 리용하여 구조화한 장면을 보여 주었다. 다각형의 변들을 펴고 적당한 면색갈을 주기 위하여 면실감처리알고리듬을 적용하였다.

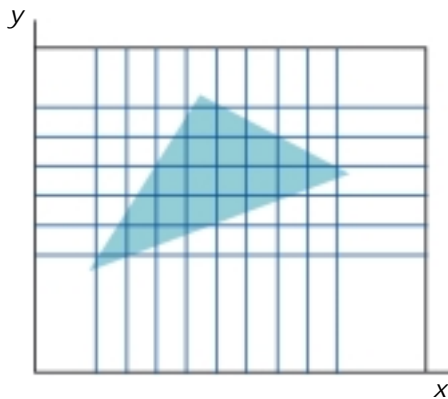



그림 10-92. 3각형조종면을 지면 격자에로 투영

그림 10-93. 우연중점변위법과 지면우의 평면형태의 조종면에 의하여 모형화한 합성장면(지형, 물, 구름의 면특징들을 개별적으로 모형화하고 실감처리한 다음 합성하였다.)

자기두제곱프락탈

프락탈물체를 만드는 다른 방법은 복소수공간의 점에 변환함수를 반복적용하는 방법이다. 2차원

적인 복소수는 $z=x+iy$와 같이 표현된다. 여기서 x와 y는 실수, $i^2=-1$이다. 3차원공간과 4차원공간에서 점은 4원수로 표현된다. 복소두제곱함수 $f(z)$는 z^2계산을 포함하며 자기두제곱함수를 리용하면 프락탈형태를 만들수 있다.

자기두제곱함수를 반복적용한 결과는 반복을 위하여 선택되는 초기위치에 따라 다음의 세가지 결과(그림 10-94)중 하나로 될수 있다.

- 변환된 위치가 무한대로 발산할수 있다.
- 변환된 위치가 수렴점이라고 부르는 유한극한점에 수렴될수 있다.
- 변환된 위치가 어떤 물체의 경계에 떨어 지게 된다.

실례로 복소수평면에서 비프락탈두제곱조작 $f(z)=z^2$은 점들을 단위원과의 관계에 따라 변환한다(그림 10-95). 절대값 $|z|$가 1보다 큰 임의의 점 z는 무한대로 향하는 위치들의 순서렬로 변환된다. $|z|<1$인 점은 자리표원점쪽으로 변환된다. 원우의 점 $|z|=1$들은 원에 남는다. 어떤 함수에서 무한대로 움직이는 점들과 유한극한쪽으로 향하는 점들사이의 경계는 프락탈이다. 프락탈물체의 경계를 **쥴리아모임**(Julia set)이라고 한다.

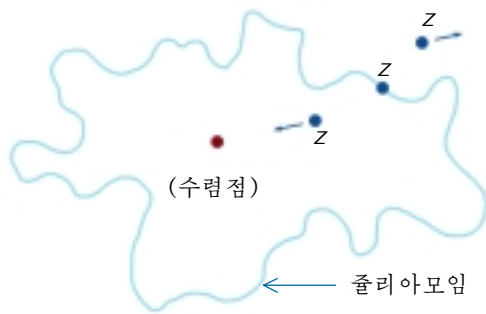



그림 10-94. 선택된 초기점의 위치에 따라 복소평면에서 자기두제곱함수 $f(z)$의 가능한 결과

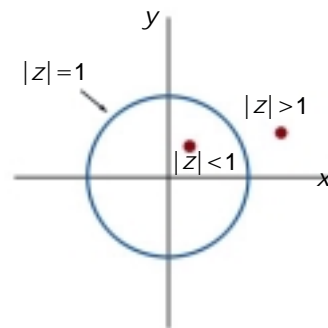



그림 10-95. 복소평면에서의 단위원(비프락탈적인 복소두제곱함수 $f(z)=z^2$에서 원안의 점은 원점으로 움직이며 한편 원밖의 점은 원에서 더 멀리로 움직인다. 원에 있는 임의의 초기점은 원에 남긴다.)

일반적으로 프락탈경계는 선택된 위치의 움직임을 검사하면 찾을수 있다. 선택된 위치가 무한대로 발산하거나 수렴점에로 수렴하면 다른 가까운 위치를 조사해 볼수 있다. 이 처리를 프락탈경계우의 위치를 찾을 때까지 반복한다. 이렇게 두제곱변환의 반복으로 프락탈형태가 만들어 진다. 복소수평면에서 함수가 간단한 경우 역변환함수를 리용하면 프락탈곡선우의 위치를 더 빨리 찾을수 있다. 곡선의 내부 또는 외부에서 선택된 초기점은 그 프락탈곡선의 위치에 수렴되게 된다(그림 10-96).

그림 10-96. 역자기두제곱함수 $z'=f^{-1}(z)$에 의하여 프락탈경계를 찾기

프락탈적인 함수는 두제곱변환함수

$$z' = f(z) = \lambda z(1-z) \qquad (10-105)$$

이다. 여기서 λ에는 임의의 복소상수값이 할당된다. 이 함수의 프락탈곡선을 찾는데 역함수방법을 리용할수 있다. 먼저 항들을 재정돈하여 2차식

$$z^2 - z + z'/\lambda = 0 \qquad (10-106)$$

을 얻는다. 그러면 역변환함수는 2차식

$$z = f^{-1}(z') = \frac{1}{2}\left(1 \pm \sqrt{1 - 4z'/\lambda}\right) \qquad (10-107)$$

이다. 복소산수연산을 리용하여 z의 실수부와 허수부에 대해 이 식을

$$x = \mathrm{Re}(z) = \frac{1}{2}\left(1 \pm \sqrt{\frac{|\mathrm{discr}| + \mathrm{Re}(\mathrm{discr})}{2}}\right)$$
$$y = \mathrm{Im}(z) = \pm\frac{1}{2}\sqrt{\frac{|\mathrm{discr}| - \mathrm{Re}(\mathrm{discr})}{2}} \qquad (10-108)$$

와 같이 푼다. 여기서 2차식의 판별식은 discr=1−4z'/ λ이다. 프락탈곡선그리기를 시작하기전에 x와 y의 몇개의 초기값(례를 들어 10개)들은 계산해 보고 버릴수 있다. 또한 이 함수는 두개의 변환된 (x, y)위치를 만들수 있기때문에 Im(discr)≥0일 때에는 매 반복걸음에서 +와 −부호를 우연적으로 선택할수 있다. Im(discr)<0일 때에는 두가지 가능한 위치들이 2번째와 4번째 4분구에 있다. 이 경우에는 x와 y가 반대부호를 가져야 한다. 다음의 프로그람은 이 자기두제곱함수의 실현으로서 그림 10-97의 두개의 실례곡선을 현시한다.

ㄱ) ㄴ)

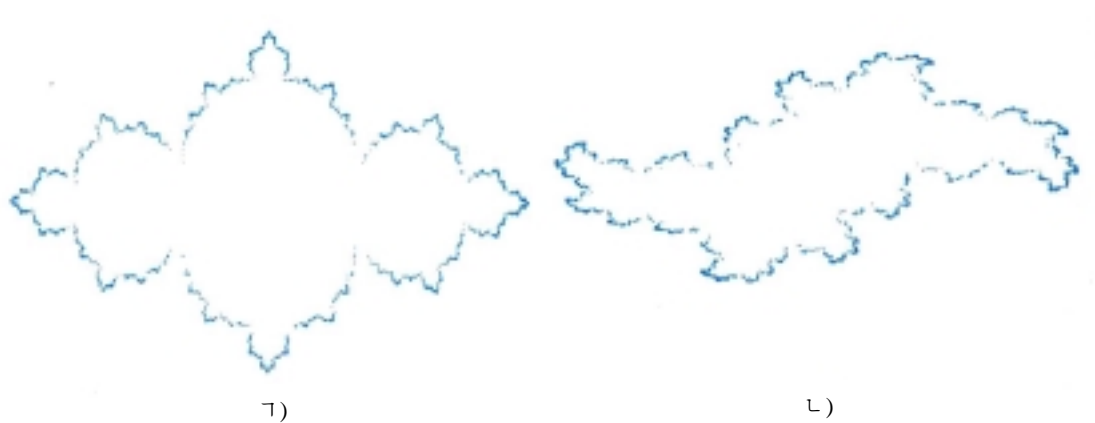



그림 10-97. 절차 selfSquare에서 함수 $f(z)=\lambda z(1-z)$의 역함수로 만든 두개의 프락탈곡선(ㄱ-λ=3, ㄴ-λ=2+i, 매개 곡선은 10,000 개의 점으로 그려 졌다.)

```
#include <math.h>
#include <values.h>
#include "graphics.h"

typedef struct {
  float x, y;
} Complex;

void calculatePoint (Complex lambda, Complex * z)
{
  float lambdaMagSq, discrMag;
```

```
  Complex discr;
  static Complex fourOverLambda = { 0, 0 };
  static BOOL firstPoint = TRUE;

  if (firstPoint) {
    /* Compute 4 divided by lambda */
    lambdaMagSq = lambda.x * lambda.x + lambda.y * lambda.y;
    fourOverLambda.x = 4 * lambda.x / lambdaMagSq;
    fourOverLambda.y = -4 * lambda.y / lambdaMagSq;
    firstPoint = FALSE;
  }
  discr.x = 1.0 - (z->x * fourOverLambda.x - z->y * fourOverLambda.y);
  discr.y = z->x * fourOverLambda.y + z->y * fourOverLambda.x;
  discrMag = sqrt (discr.x * discr.x + discr.y * discr.y);

  /* Update z, checking to avoid the sqrt of a negative number */
  if (discrMag + discr.x < 0)
    z->x = 0;
  else
    z->x = sqrt ((discrMag + discr.x) / 2.0);
  if (discrMag - discr.x < 0)
    z->y = 0;
  else
    z->y = 0.5 * sqrt ((discrMag - discr.x) / 2.0);

  /* For half the points, use negative root, placing point in quadrant 3 */
  if (random() < MAXINT/2) {
    z->x = -z->x;
    z->y = -z->y;
  }

  /* When imaginary part of discriminant is negative, point
     should lie in quadrant 2 or 4, so reverse sign of x */
  if (discr.y < 0) z->x = -z->x;

  /* Finish up calculation for the real part of z */
  z->x = 0.5 * (1 - z->x);
}

void selfSquare (Complex lambda, Complex z, int count)
{
  int k;

  /* Skip the first few points */
  for (k=0; k<10; k++)
    calculatePoint (lambda, &z);
  for (k=0; k<count; k++) {
    calculatePoint (lambda, &z);
    /* Scale point to fit window and draw */
    pPoint (z.x*WINDOW_WIDTH, 0.5*WINDOW_HEIGHT+z.y*WINDOW_HEIGHT);
  }
```

```
    }
```

$|\lambda|$=1일 때 변수가 x, y, λ인 자기두제곱함수 $f(z)=\lambda z(1-z)$의 3차원모양을 그림 10-98에서 보여주었다. 이 그림의 매개 자름면은 복소수평면에서의 프락탈곡선이다.

대단히 유명한 프락탈형태는 두제곱변환에서 발산하지 않는 복소값 z들의 모임인 **만델브로트모임**(Mandelbrot set)으로부터 얻어 진다.

$$z_0 = z$$
$$z_k = z_{k-1}^2 + z_0, \qquad k = 1, 2, 3, \cdots \qquad (10-109)$$

그림 10-98. 수직축이 정규화된 λ인 3차원적으로 표시된 함수 $f(z)=\lambda z(1-z)$

즉 처음에 복소수평면에서 점 z를 선택한 다음 변환위치 z^2+z를 계산한다. 그다음 걸음에서 이 변환된 위치를 두제곱하고 본래의 z값을 더한다. 이 절차를 변환이 발산하는가, 안하는가를 결정할수 있을 때까지 반복한다. 복소수평면에서 수렴구역의 경계는 프락탈이다.

식 10-109의 변환을 실현하기 위하여 먼저 복소수평면에서 창문을 선택한다. 이 창문안의 위치들은 그다음 선택된 화면보임창의 색부호화된 화소위치에 넘겨진다(그림 10-99). 화소의 색은 식 10-109를 리용하여 변환할 때 복소수평면의 대응하는 점의 발산속도에 따라 선택된다. 만일 복소수의 절대값이 2보다 크면 그 점은 이 자기두제곱조작에서 빨리 발산하게 된다. 따라서 두제곱조작은 복소수의 절대값이 2를 넘거나 또는 미리 설정된 반복회수에 도달할 때까지 반복할수 있다. 반복의 최대수는 보통 100과 1000사이의 어떤 값으로 설정하며 더 낮은 값은 계산속도를 높이는데 리용될수 있다. 그러나 반복한계를 낮게 설정하면 수렴구역의 경계(쥴리아모임)에서 세부를 약간 잃어 버리는 경향이 있다.

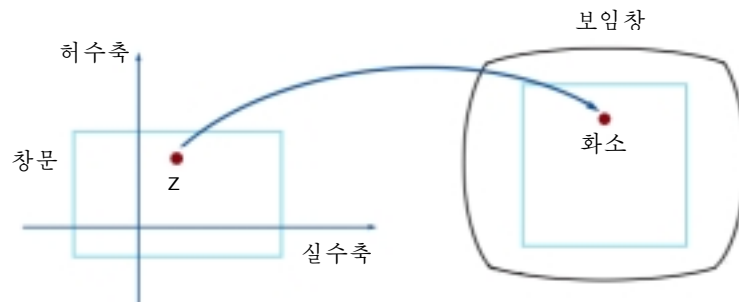



그림 10-99. 복소수평면의 위치를 현시장치의 색부호화된 화소위치에로 넘기기

반복순환의 마지막에 실행한 반복회수에 따라 색값을 선택한다. 실례로 반복회수가 최대값이면 화소를 검은색으로 하고 반복회수가 0에 가까우면 화소를 붉은색으로 할수 있다. 그러면 다른 색값

들은 0부터 최대값사이의 반복회수값에 따라 선택할수 있다. 각이한 색입히기를 선택한다면 만델브로트모임에 대한 여러가지 멋 있는 현시를 얻을수 있다. 이 모임에 대한 색부호화의 한가지를 그림 10-100 ㄱ에서 보여 주었다.

만델브로트모임을 현시하는 알고리듬을 다음의 프로그람에 주었다. 대부분의 모임은 복소수평면의 다음의 구역안에 포함된다.

$$-2.25 \leq \mathrm{Re}(z) \leq 0.75$$

$$-1.25 \leq \mathrm{Im}(z) \leq 1.25$$

더 작은 창문구역을 련속적으로 선택하는 방법으로 현시구역을 확대해 봄으로써 모임의 경계를 따라 세부를 조사할수 있다. 그림 10-100에서는 만델브로트모임의 색부호화된 모양과 이 주목할만한 모임의 일부 특징들을 설명하는 련속적인 확대상을 보여 주었다.

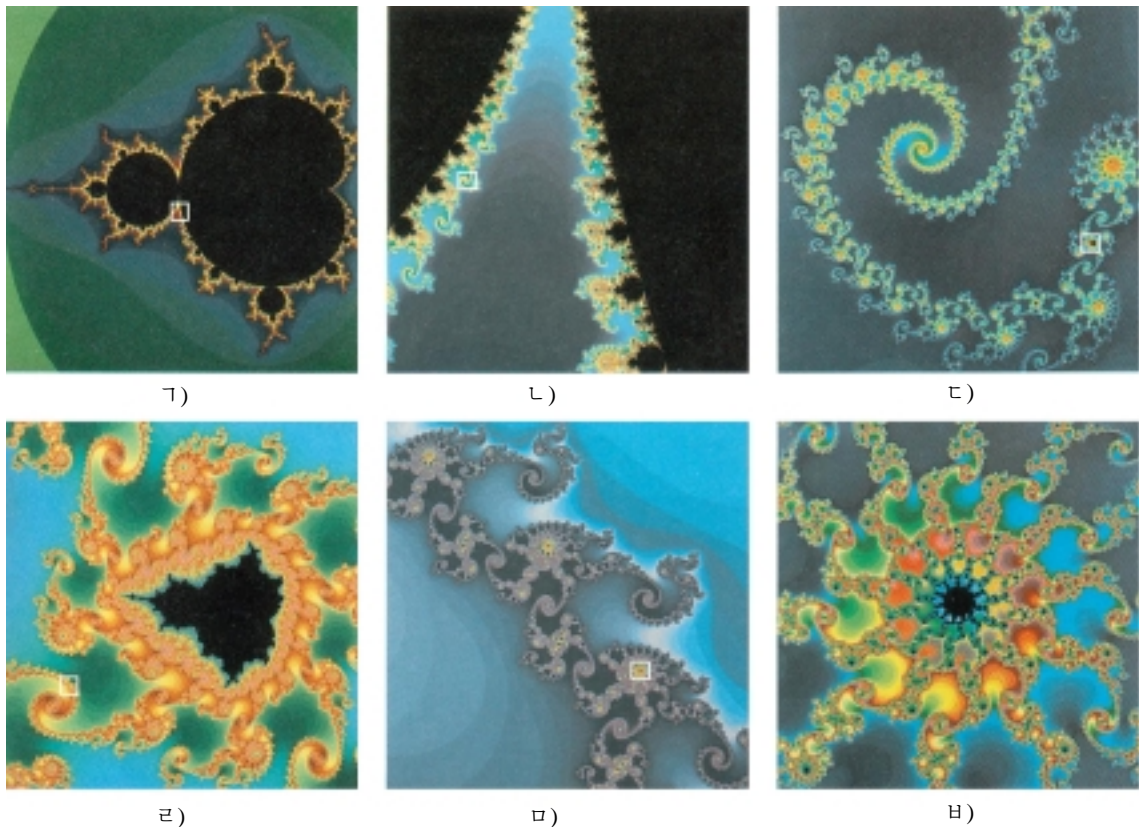



그림 10-100. 만델브로트모임을 확대하여 보기(만델브로트모임의 ㄱ현시에서 시작하여 선택되는 구역들을 ㄴ로부터 ㅂ까지 확대하여 본다. 흰 직4각형륜곽선은 매개 련속되는 확대보기를 위하여 선택되는 창문구역이다.)

```
#include "graphics.h"

typedef struct { float x, y; } Complex;

Complex complexSquare (Complex c)
{
  Complex cSq;
  cSq.x = c.x * c.x - c.y * c.y;
  cSq.y = 2 * c.x * c.y;
  return (cSq);
```

```
}

int iterate (Complex zInit, int maxIter)
{
  Complex z = zInit;
  int cnt = 0;

  /* Quit when z * z > 4 */
  while ((z.x * z.x + z.y * z.y <= 4.0) && (cnt < maxIter)) {
    z = complexSquare (z);
    z.x += zInit.x;
    z.y += zInit.y;
    cnt++;
  }
  return (cnt);
}

void mandelbrot (int nx, int ny, int maxIter, float realMin,
                    float realMax, float imagMin, float imagMax)
{
  float realInc = (realMax - realMin) / nx;
  float imagInc = (imagMax - imagMin) / ny;
  Complex z;
  int x, y;
  int cnt;

  for (x=0, z.x=realMin; x<nx; x++, z.x+=realInc)
    for (y=0, z.y=imagMin; y<ny; y++, z.y+=imagInc) {
      cnt = iterate (z, maxIter);
      if (cnt == maxIter)
        setColor (BLACK);
      else
        setColor (cnt);
      pPoint (x, y);
    }
}
```

식 10-105와 같은 복소함수변환은 프락탈면과 프락탈립체를 만드는데로 확장할수 있다. 이런 물체들을 만들 때에는 3차원 및 4차원공간에서의 점의 변환을 위한 4원수표현을 리용한다(부록 6). 4원수는 복소수평면에서의 수개념의 확장으로서 4개의 요소 즉 하나의 실수부와 3개의 허수부를 가지고 표현할수 있다.

$$q = s + ia + jb + kc \qquad (10-110)$$

여기서 $i^2 = j^2 = k^2 = -1$이다. 실수부 s를 4원수의 스칼라부라고 하며 허수부들은 4원수의 벡토르부 $\mathbf{v}=(a, b, c)$라고 한다.

부록 6에서 설명되는 4원수의 곱하기와 더하기규칙을 리용하면 자기두제곱함수와 다른 반복방법들을 프락탈곡선이 아니라 프락탈물체의 면을 만드는데 적용할수 있다. 기본절차는 프락탈물체안의

위치에서 시작하여 외부(발산)점이 식별될 때까지 련속적으로 점들을 만드는것이다. 처음에 내부점은 면의 점으로 보유된다. 다음 이 면점의 근방을 내부(수렴) 또는 외부(발산)인가를 결정하기 위하여 검사한다. 외부점에 련결되는 임의의 내부점은 면의 점이다. 이 방법의 절차는 점들을 면으로부터 너무 멀리 떨어 지지 않고 프락탈경계를 따라 가게 한다. 4차원프락탈이 만들어 졌을 때 3차원자름면은 현시장치의 2차원면에 투영된다.

4차원공간에서 자기두제곱프락탈을 만드는 절차는 반복함수를 평가하고 점들을 검사하는데 상당한 계산시간을 소비한다. 면의 매개 점은 면의 내부와 외부한계를 주는 작은 바른6면체로 표현할수 있다. 프락탈의 3차원투영에 대한 이러한 프로그람으로부터의 출력은 일반적으로 면의 바른6면체들에 대한 몇백만개의 정점들을 포함한다. 프락탈물체의 현시는 매개 면의 바른6면체에 대하여 빛과색을 결정하는 조명모형을 적용하여 실현한다. 다음에 보이지 않는 면방법이 물체의 보이는 면들만 현시되도록 하기 위하여 적용된다. 그림 10-101과 10-102에서는 3차원에 투영된 자기두제곱4차원프락탈의 실례를 보여 주었다.

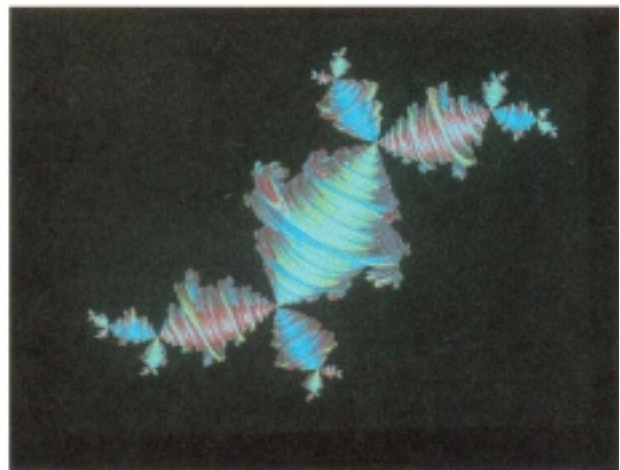
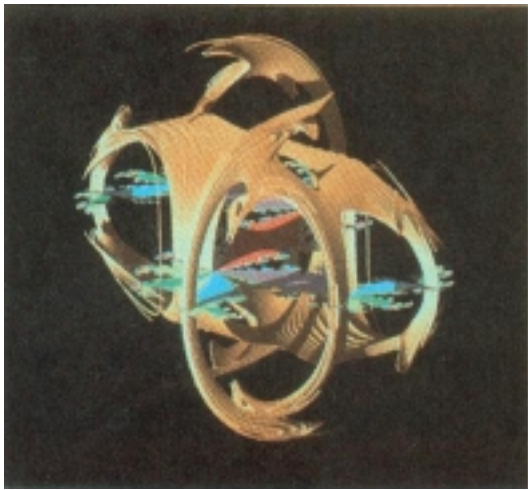



그림 10-101. 자기두제곱4원수함수 $f(q)=\lambda q(1-q)$에 의하여 만든 4차원 프락탈의 3차원투영(ㄱ-$\lambda=1.475+0.9061i$, ㄴ-$\lambda=-0.57+i$)

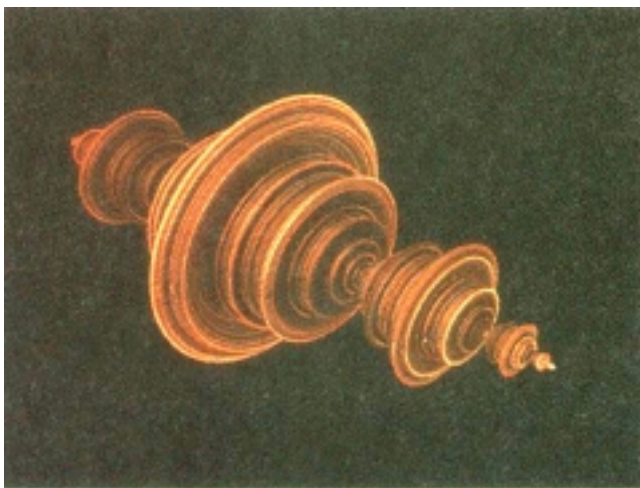



그림 10-102. 자기두제곱4원수함수 $f(q)=q^2-1$에 의하여 만든 4차원물체의 3차원면투영

자기역프락탈

프락탈형태를 만드는데 여러가지 기하학적인 역변환들을 리용할수 있다. 점들의 초기모임에서 시작하여 초기점들을 프락탈로 변환하기 위하여 비선형역연산을 반복적으로 적용하자.

실례로 반경이 r이고 중심이 위치 $\mathbf{P}_0=(x_0, y_0)$에 있는 원에 대한 2차원역변환을 고찰한다. 원밖의

임의의 점 **P**는 변환

$$(\overline{\mathbf{P}_0\mathbf{P}})(\overline{\mathbf{P}_0\mathbf{P}'}) = r^2 \qquad (10-111)$$

에 의하여 원안의 위치 **P'**에로 반전된다(그림 10-103). 반대로 이 변환은 원안의 임의의 점을 원밖의 점으로 반전시킨다. **P**와 **P'**는 다같이 원의 중심 $\mathbf{P}_0$을 통과하는 직선우에 있다.

두 점의 자리표를 각각 **P**=(x, y), **P'**=(x', y')라고 하면 식 10-111을

$$[(x-x_0)^2+(y-y_0)^2]^{1/2}[(x'-x_0)^2+(y'-y_0)^2]^{1/2}=r^2$$

와 같이 쓸수 있다.

또한 두 점은 원의 중심을 통과하는 직선우에 있기때문에 $(y-y_0)/(x-x_0)=(y'-y_0)/(x'-x_0)$이다. 따라서 변환된 자리표값은

$$x' = x_0 + \frac{r^2(x-x_0)}{(x-x_0)^2+(y-y_0)^2}, \quad y' = y_0 + \frac{r^2(y-y_0)}{(x-x_0)^2+(y-y_0)^2} \qquad (10-112)$$

이다.

그림 10-104에서는 다른 원둘레우의 점들의 반전을 보여 주었다. 반전되는 원은 $\mathbf{P}_0$을 통과하지 않는 한 다른 원으로 변환된다. 그러나 원둘레가 $\mathbf{P}_0$을 통과하면 그 원은 직선으로 변환된다. 반대로 $\mathbf{P}_0$을 통과하지 않는 직선우의 점들은 원으로 반전된다. 그러므로 직선은 역변환에서 불변이다. 또한 참조원과 직교하는 원도 이 변환에서 불변이다. 즉 두 원의 접선은 사귐점에서 수직이다.

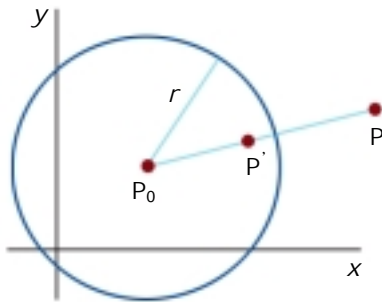



그림 10-103. 점 **P**를 반경 *r*인 원안의 위치 **P'**에로 반전

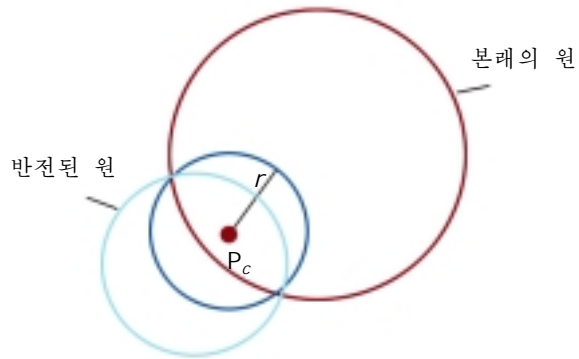



그림 10-104. 한 원을 다른 원에 대하여 반전

원들의 모임에서 시작하여 서로 다른 참조원을 리용하는 변환을 반복적으로 적용하는 방법으로 이 역변환에 의해 여러가지 프락탈형태들을 만들수 있다. 류사하게 직선모임에 대하여 원의 반전을 적용할수 있다. 류사한 반전방법을 다른 물체들에 대하여서도 얻을수 있다. 그리고 절차를 구, 평면, 3차원공간의 다른 형태들에 일반화할수 있다.

19절. 형태문법과 기타 수속적인 방법

물체의 세부를 만들기 위한 기타 수속적인 방법들이 많이 개발되였다. **형태문법**(shape grammar)은 초기물체에 본래의 형태에 맞는 세부도를 추가하기 위하여 적용할수 있는 생성규칙들의 모임이다. 이 변환은 물체의 기하(형태)를 변경시킬수도 있으며 면의 색이나 면의 걸문양세부를 추가하는데도 적용할수 있다.

생성규칙들의 모임이 주어 지면 형태설계자는 다음에 주어 진 초기물체로부터 마지막구조물에로 변환의 매 걸음에서 서로 다른 규칙들을 적용하여 볼수 있다. 그림 10-105에서는 3각형형태를 변경시키는 4개의 기하학적치환규칙을 보여 주었다. 이 규칙들에 대한 기하변환은 생성규칙편집기가 있는 체계에서 그려 지는 입력그림에 기초하여 알고리듬적으로 씌여 질수 있다. 즉 매개 규칙은 초기 및 마지막형태를 보여 주는 방법으로 도형적으로 표현된다. 다음 Mathematica 또는 도형처리능력이 있는 기타 다른 프로그람작성언어들에 의해 실현될수 있다.

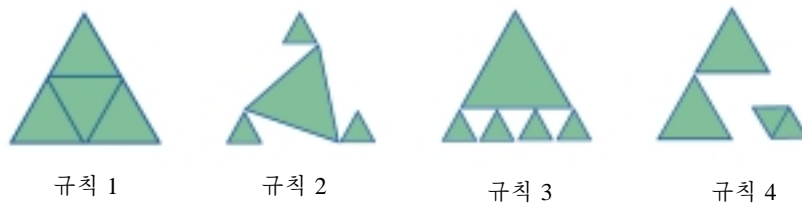



그림 10-105. 바른3각형의 형태를 부분분할 및 변경하는 4개의 기하학적치환규칙

그림 10-105의 기하학적치환들의 응용을 그림 10-106에서 보여 주었다. 여기서 그림 10-106 ㄹ는 그림 10-106 ㄱ의 초기3각형에서 시작하여 4개의 규칙을 련속적으로 적용하면 얻어 진다. 그림 10-107에서는 3각형치환규칙에 의하여 만들어 지는 다른 형태를 보여 주었다.

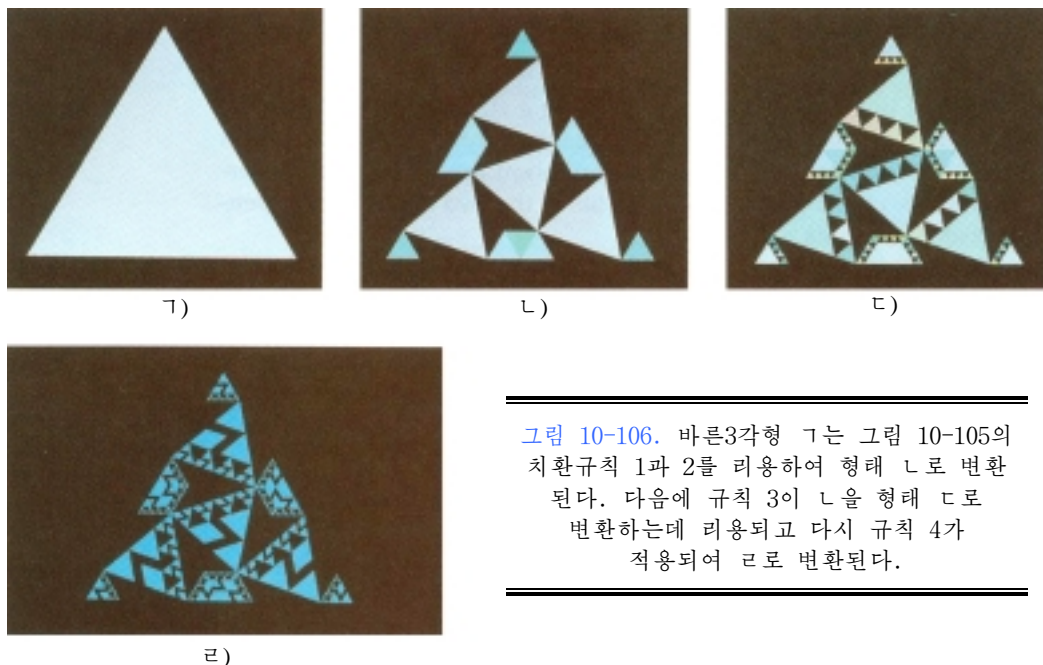



그림 10-106. 바른3각형 ㄱ는 그림 10-105의 치환규칙 1과 2를 리용하여 형태 ㄴ로 변환된다. 다음에 규칙 3이 ㄴ을 형태 ㄷ로 변환하는데 리용되고 다시 규칙 4가 적용되여 ㄹ로 변환된다.

그림 10-107. 3각형형태를 변화시키는 기하학적 치환규칙에 의하여 만들어 진 설계

류사한 조작에 의하여 3차원형태 및 면특징들을 변환한다. 그림 10-108에서는 다면체에 적용된 기하학적치환의 결과를 보여 주었다. 그림 10-109에 보여 준 물체들의 초기형태는 20면체 즉 20개의 면을 가지는 다면체이다. 기하학적치환은 20면체의 면들에 적용되였으며 결과를 다각형정점들을 둘러 싸는 구면에 투영하였다.

그림 10-108. 프리즘형태를 변화시키는 기하학적치환규칙에 의하여 만들어 진 설계(이 설계에 대한 초기형태는 Rubik의 뱀의 표현이였다.)

그림 10-109. 3각형치환규칙을 20면체의 면들에 적용한 후 결과를 구면에 투영하여 만든 설계

물체형태를 표현하기 위하여 생성규칙을 리용하는 다른 실례는 L문법 또는 접붙이기이다. 이 규칙들은 식물을 표현하는 방법을 준다. 실례로 나무의 형태는 가지와 잎사귀들이 붙어 있는 줄기로 표현할수 있다. 그러므로 나무는 개별적인 가지에서의 가지 및 잎사귀들의 매개 련결을 규정하는 규칙에 의하여 모형화할수 있다. 다음에 물체의 구조들을 개별적인 자리표위치에 놓으면 기하학적인 표현이 주어 진다.

그림 10-110에서는 상품화된 식물만들기프로그람에 의하여 만든 여러가지 식물들과 나무들이 들어 있는 장면을 보여 주었다. 식물을 만드는 쏘프트웨어의 절차들은 식물학법칙에 기초한다.

그림 10-110. TDI-AMAP쏘프트웨어패키지에 의하여 만들어 진 현실감 있는 풍경(그것은 식물학법칙에 기초한 절차들을 리용하여 100여가지의 식물과 나무를 만들수 있다.)

20절. 립자계

립자계(particle system)는 류체와 같은 특성을 나타내는 자연적인 물체, 다른 불규칙적인 형태의 물체들을 모형화하는 방법이다. 이 방법은 류동, 물결치기, 흩어짐, 팽창에 의하여 시간에 따라 변하는 물체를 표현하는데 아주 좋다. 이런 특성을 가지는 물체에는 구름, 연기, 불길, 불꽃, 폭포, 물보라, 수풀이 있다. 실례로 립자계는 영화 Star Trek II-The Wrath of Khan에서 《폭탄》에 의한 행성의 폭발과 팽창되는 화염장벽을 모형화하는데 리용되였다.

공간의 어떤 정해 진 구역안에서 물체를 발생시키고 시간에 따라 파라메터들을 변화시키는데는 우연절차를 리용한다. 매개 물체는 어떤 우연시간후에 지워 진다. 립자의 경로와 면의 특징들은 수명시간안에서 색부호화되여 현시되게 된다.

립자의 형태는 작은 구, 타원체, 직6면체, 기타 형태로 될수 있다. 립자의 크기와 형태는 시간에 따라 우연적으로 변할수 있다. 또한 립자의 투명도, 색, 이동과 같은 다른 특성들도 모두 우연적

으로 변할수 있다. 일부 응용들에서 립자의 운동은 인력마당과 같은 지적되는 힘에 의하여 조종될수 있다.

매 립자가 움직일 때 그의 경로는 개별적인 색으로 그려 지고 현시된다. 실례로 불꽃무늬는 그림 10-111에서와 같이 립자들을 구모양의 공간구역안에서 우연적으로 발생시키고 그것들을 방사형으로 바깥쪽으로 움직이게 하여 현시할수 있다. 립자의 경로는 폭발하는 립자의 온도를 모의하기 위하여 실례로 붉은색으로부터 누런색으로 색부호화할수 있다. 류사하게 수풀의 현실감 있는 현시는 땅에서 우로 쏘고 중력에 의하여 지구로 다시 떨어 지는 《탄도》립자(그림 10-112)에 의하여 모형화되였다. 이 경우에 립자의 경로는 바닥이 좁은 원기둥안에서 발생되고 풀색으로부터 누런색으로 색부호화될수 있다.

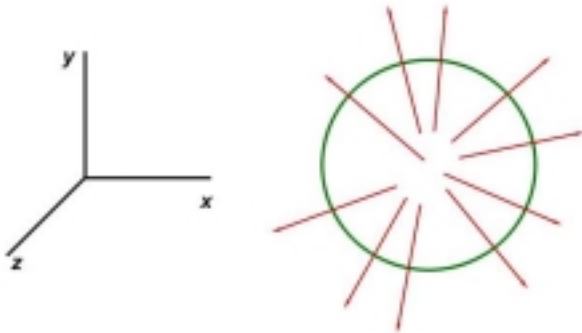



그림 10-111. 구의 중심으로부터 바깥쪽으로 방사형으로 움직이는 립자들을 가지는 립자계에 의한 불꽃의 모형화

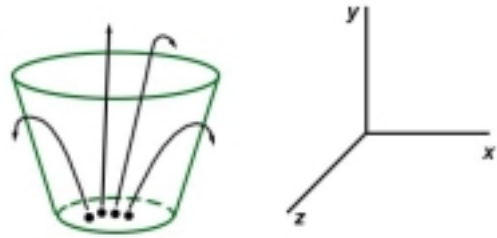



그림 10-112. 바닥이 좁은 원기둥안에서 웃쪽으로 립자들을 방출시켜 풀밭을 모형화(립자의 경로는 중력때문에 포물선이다.)

그림 10-113에서는 립자계에 의한 폭포의 모의를 보여 주었다. 물립자는 고정된 높이에서 떨어지면서 장애물에 의하여 방향이 바뀌여 지고 그다음 바닥으로부터 우로 튀여 난다. 매 단계에서의 립자의 경로를 구별하기 위하여 서로 다른 색들을 리용한다. 물체의 분해를 모의하는 동화실례를 그림 10-114에 보여 주었다. 왼쪽의 물체는 오른쪽의 립자분포로 분해된다. 여러가지 표현들에 의하여 형성된 합성장면을 그림 10-115에 보여 주었다. 장면은 립자계로 만든 풀밭, 프락탈산, 걸문양입히기, 기타 면실감처리절차를 리용하여 모형화하였다.

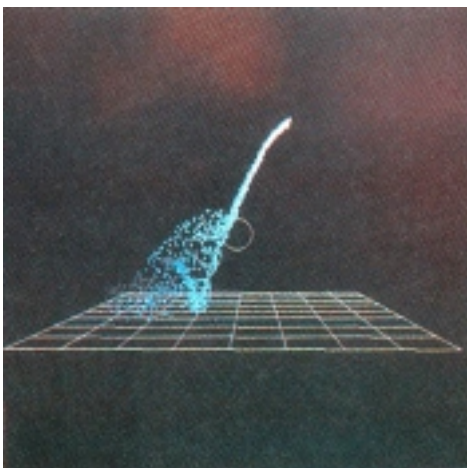



그림 10-113. 돌(동그라미)을 때리는 폭포의 상태모의(물립자들은 돌에 의하여 편향되며 바닥으로부터 우로 튀여 난다.)

그림 10-114. 립자들의 구름으로 분해되는 물체

그림 10-115. 립자계로 만든 풀, 프락탈산, 걸문양 입혀 진 면을 보여 주는 Road to Point Reyes라는 이름이 붙은 장면

21절. 물리학적모형화

바줄, 천쪼각, 연한 고무공같은 딴딴하지 않은 물체는 물체의 성질을 외부 및 내부힘들의 호상작용에 의하여 표현하는 **물리학적모형화**(physically based modelling)방법으로 표현할수 있다. 의자뒤에 드리운 뜨개직물수건의 형태에 대한 정밀한 표현은 천안의 뜨개고리에 미치는 의자의 영향과 실들사이의 호상작용을 고찰하여 얻는다.

딴딴하지 않은 물체를 모형화하는 일반적인 방법은 물체를 유연하게 련결되는 점마디들의 그물로 근사화하는것이다. 련결의 한가지 간단한 형태는 용수철이다. 그림 10-116에 고무판의 성질을 근사화하는데 리용될수 있는 2차원용수철그물의 부분을 보여 주었다. 이런 용수철그물은 3차원적으로 설정되여 고무공 또는 묵 같은 덩어리를 모형화할수 있다. 동질물체에서는 그물에 같은 용수철들을 리용할수 있다. 물체가 서로 다른 방향에서 서로 다른 특성을 가지게 하려면 서로 다른 방향에서 서로 다른 특성의 용수철을 리용할수 있다. 외부힘이 용수철그물에 작용할 때 개별적인 용수철들에 대한 잡아 당김 또는 압축은 용수철의 억세기라고도 하는 용수철상수 k의 값에 관계된다.

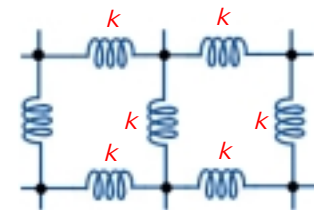



그림 10-116. 같은 용수철상수 k로 만든 2차원용수철그물

힘 F_x의 영향하에서 마디위치의 수평변위 x를 그림 10-117에 보여 준다. 용수철을 지나치게 당기지 않는 한 균형위치로부터의 변위량 x는 후크법칙

$$F_s = -F_x = -kx \qquad (10-113)$$

를 리용하여 정확하게 근사화할수 있다. 여기서 F_s는 당겨 지는 마디에서 크기가 같고 방향이 반대인 용수철의 튕성힘이다. 이 관계는 용수철의 x만한 수평압축에 대해서도 성립한다. y 및 z방향에서의 변위와 힘성분들에서도 류사한 관계를 가진다.

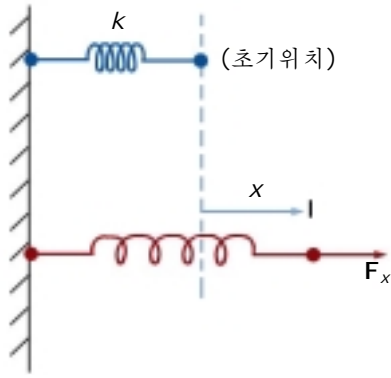



그림 10-117. 한 끝이 고정된 용수철을 다른 끝에서 잡아 당기는 외부힘 F_x

만일 물체가 완전한 륀성체이면 외부힘이 제거될 때 물체는 본래의 형태로 돌아 간다. 그러나 빠데(putty)나 기타 일부 변형될수 있는 물체를 모형화하려 한다면 외부힘이 제거될 때 용수철이 자기의 본래형태로 돌아 가지 않도록 용수철의 특성을 수정하여야 한다. 그러면 가해지는 힘들이 달라 질 때 물체를 약간 다르게 변형시킬수 있다. 용수철을 리용하지 않고 마디들사이 련결을 륀성재료로 모형화할수도 있으며 외부힘의 영향하에서 물체의 형태를 결정하기 위하여 변형에네르기함수를 최소화한다. 이 방법은 천에 대한 더 좋은 모형을 주는데 서로 다른 천재료의 성질을 표현하기 위한 여러가지 에네르기함수들이 발명되였다.

딴딴하지 않은 물체를 모형화하기 위하여서는 먼저 물체에 작용하는 외부힘을 설정한다. 다음에 물체를 표현하는 그물을 통한 힘의 전파를 고찰한다. 이것은 그물을 따라서 마디들의 변위를 결정하기 위하여 풀어야 하는 련립방정식들의 모임을 준다.

그림 10-118에 용수철그물로 모형화한 바나나껍질을 보여 주었으며 그림 10-119의 장면은 에네르기함수를 리용하여 모형화하고 걸문양을 입힌 천의 실례들을 보여 준다. 에네르기함수계산을 리용하여 그물의 파라메터들을 조종하면 서로 다른 종류의 천을 모형화할수 있다. 그림 10-120에 탁에 드리운 면직물, 모직물, 폴리에스테르면직물재료에 대한 모형들을 보여 주었다.

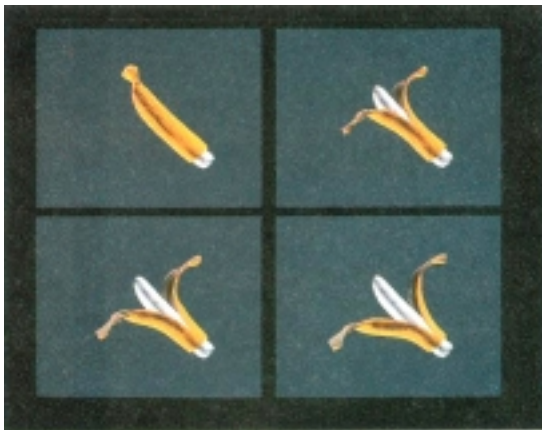



그림 10-118. 바나나껍질의 유연한 움직임을 용수철그물에 의하여 모형화

그림 10-119. 가구에 드리운 천의 유연한 움직임을 에네르기함수최소화를 리용하여 모형화

ㄱ)

ㄴ)

ㄷ)

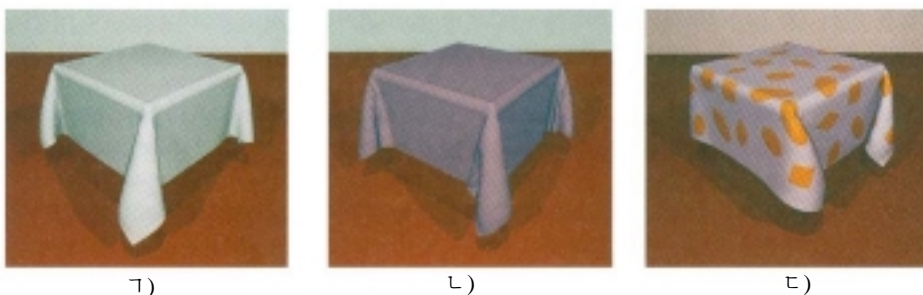



그림 10-120. 에네르기함수최소화를 리용한 면직물(ㄱ), 모직물(ㄴ), 폴리에스테르면직물(ㄷ)특성의 모형화

물리학적모형화방법들을 동화에 적용하면 운동경로를 보다 정밀하게 표현할수 있다. 과거에 동화는 흔히 스플라인경로와 운동학을 리용하여 지적하였으며 운동파라메터들은 위치와 속도에만 기초하여 결정하였다. 물리학적모형화는 힘과 가속도를 포함하는 동력학방정식을 리용하여 운동을 표현한다. 동력학방정식에 기초한 동화표현은 운동학방정식에 기초한것보다 더 현실감 있는 운동을 만든다.

22절. 자료모임의 가시화

도형적인 방법들을 과학 및 공학해석에서 방조수단으로 리용하는것을 일반적으로 과학적가시화라고 한다. 여기에는 도형적인 방법이 없이는 해석하기가 힘들거나 불가능한 자료모임과 처리들의 가시화가 속한다. 실례로 가시화기술은 초대형콤퓨터, 위성과 우주비행선의 스캐너, 전파천체망원경, 의학화상스캐너와 같은 방대한 규모의 자료원천의 출력을 처리하는데 필요하다. 콤퓨터모의의 수값풀이와 관찰설비로부터 자주 수백만개의 자료점들이 발생되며 생자료를 간단히 주사하여 경향과 관계를 결정하는것은 힘들다. 류사하게 가시화기술은 긴 시간주기에서 일어 나거나 또는 량자력학현상 및 빛에 가까운 속도로 움직이는 물체에 의하여 만들어 지는 특수상대성효과와 같은 직접 관찰할수 없는 과정들을 해석하는데 쓸모 있다. 과학적가시화는 콤퓨터도형처리, 화상처리, 콤퓨터시각, 자료들을 더 잘 리해하도록 정보를 시각적으로 현시, 강조, 처리하는 다른 분야들의 방법들을 리용한다. 상업, 공업, 다른 비과학분야들에서 쓰이는 류사한 방법을 때때로 업무가시화라고 한다.

자료모임은 그것들의 공간적인 분포와 자료의 류형에 따라 분류된다. 2차원자료모임은 면에 분포되는 값들을 가지며 3차원자료모임은 바른6면체, 구 또는 기타 다른 공간구역의 내부에 분포되는 값들을 가진다. 자료의 류형에는 스칼라, 벡토르, 텐소르, 다변량자료가 속한다.

스칼라마당의 시각적인 표현

스칼라량은 단일값을 가지는 량이다. 스칼라자료모임은 공간적인 위치뿐아니라 시간에 분포될수 있는 값들을 포함한다. 또한 자료값은 다른 스칼라파라메터의 함수일수 있다. 물리적인 스칼라량의 몇가지 실례는 에네르기, 밀도, 질량, 온도, 압력, 전하, 저항, 반사률, 주파수, 물함유량이다.

스칼라자료모임을 가시화하는 일반적인 방법은 자료값들의 분포를 위치 및 시간과 같은 다른 파라메터들의 함수로 보여 주는 그라프 또는 도표를 리용하는것이다. 자료가 면에 분포되면 자료값들을 면에서 우로 오르는 수직막대기로 현시할수 있으며 원활한 면을 현시하기 위하여 자료값들을 보간할수 있다. 또한 스칼라자료모임의 서로 다른 값들을 구별하는데는 허위색방법이 리용되며 색부호화기술은 그라프 및 도표방법과 결합될수 있다. 스칼라자료모임을 색부호화하기 위하여 색들의 범위를 선택하고 자료값들의 범위를 색범위로 넘긴다. 실례로 푸른색은 제일 낮은 스칼라값에 할당될수 있으며 붉은색은 제일 높은 값에 할당될수 있다. 그림 10-121에서는 색부호화된 면도면의 실례를 보여 주었다. 자료모임의 색부호화는 까다로울수 있다. 왜냐하면 일부 색결합들은 자료를 잘못 리해하게 할수 있기때문이다.

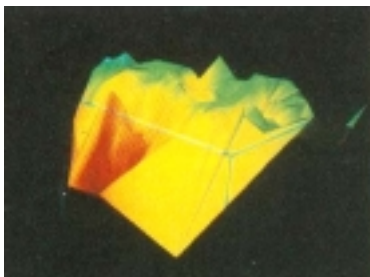



그림 10-121. 1987. 10. 증권시장붕괴시 주식성장 가능성을 보여 주는 재정면도(붉은색은 높은 수입을 지적한다. 그림은 낮은 성장주식이 더 잘 붕괴되였다는것을 보여 준다.)

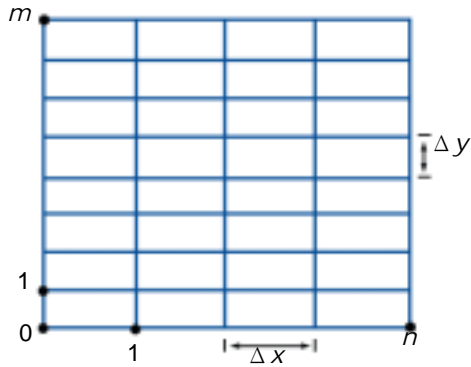



그림 10-122. 격자선들의 사귐점에서의 자료값들을 가지는 규칙적인 2차원격자(x 격자선은 일정한 Δx간격을 가지며 y격자선은 일정한 Δy간격을 가진다. 여기서 x와 y방향의 간격은 같지 않을수 있다.)

등고선도는 면에 분포된 자료모임에 대한 등값선(상수스칼라값의 선)을 현시하는데 리용된다. 등값선들은 공간구역에서 자료값들의 범위와 변화를 보여 주기 위하여 일부 편리한 구간으로 간격 띄여 진다. 대표적인 응용은 바다면우의 높이의 등고선도이다. 보통 등고선방법은 그림 10-122에서와 같이 규칙적인 격자에 분포되는 자료값들의 모임에 적용된다. 규칙적인 격자는 등간격격자선을 가지며 격자사귐점들에서 자료값을 안다. 콤퓨터모의의 수값풀이는 보통 규칙적인 격자에서의 자료분포를 만들도록 설정되며 한편 관찰되는 자료모임은 자주 불규칙적으로 간격 띄여 진다. 등고선방법은 여러가지 종류의 불규칙적인 격자들에 대하여 창안되였지만 불규칙적인 자료분포는 자주 규칙적인 격자로 변환된다. 2차원등고선알고리듬은 어느 세포변두리가 개개 등값선에 의하여 가로질러 지는가를 결정하기 위하여 격자세포의 네 구석을 검사하고 격자안의 세포마다에 등값선을 추적한다. 등값선은 보통 그림 10-123에서 설명되는바와 같이 매개 세포를 가로지르는 직선부분들로 현시된다. 때때로 등값선은 스플라인곡선으로 표시되지만 스플라인맞추기는 자료모임의 리해에서 비일관성과 오해를 가져 올수 있다. 실례로 두개의 스플라인등값선들은 사귈수 있으며 또는 등값곡선경로는 실지의 자료경향을 지적하지 않을수 있다. 왜냐하면 자료값은 세포의 구석들에서만 알고 있기때문이다. 등고선프로그람은 조사자가 비일관성을 바로 잡기 위하여 등값선을 대화식으로 조정할수 있게 한다. xy평면에서 세개의 겹치는 색부호화된 등고선도의 실례를 그림 10-124에 보여 주었으며 그림 10-125에는 불규칙적인 형태의 공간에 대한 등고선과 색부호화를 보여 주었다.

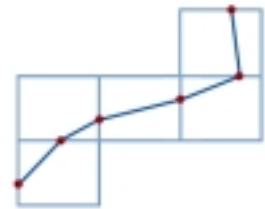



그림 10-123. 5개의 격자세포를 가로지르는 등값선의 경로

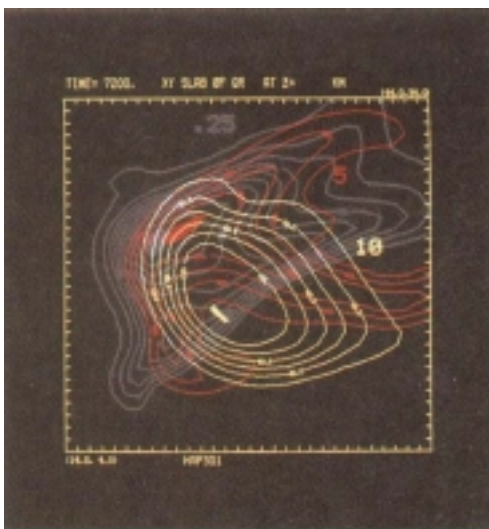



그림 10-124. xy평면의 동일한 구역안에서 세개의 자료모임의 색부호화된 등고선도

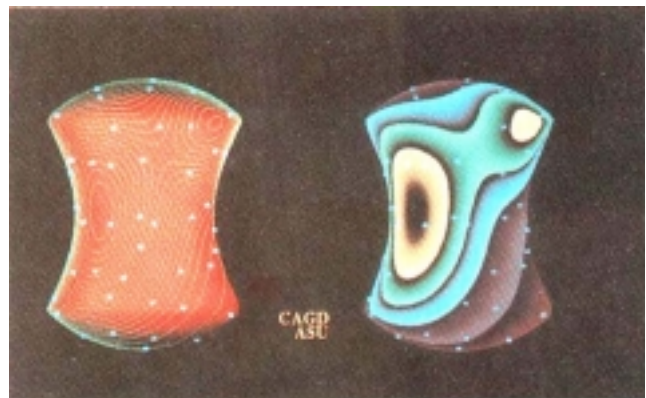



그림 10-125. 사과속모양의 공간구역의 면에서의 색부호화된 등고선도

3차원스칼라자료마당에 대하여 자름면을 취하고 자름면에서의 2차원자료분포를 현시할수 있다. 자름면에서의 자료값들을 색부호화할수 있으며 또는 등값선을 표시할수 있다. 가시화프로그람들은 일반적으로 자름면을 임의의 각도로 취할수 있는 절단루틴을 제공한다. 그림 10-126에서는 상품화된 층자르기프로그람에 의하여 만들어 진 현시를 보여 주었다.

2차원자름면을 보는 대신에 간단히 3차원등고선도인 하나 또는 그이상의 등값면들을 표시할수 있다(그림 10-127). 두개의 겹침등값면들이 표시될 때 두 등값면들의 형태를 다같이 볼수 있도록 바깥면은 투명하게 한다. 등값면을 만드는것은 3차원격자세포를 가지는것과 함께 등값면의 부분을 찾아 내기 위하여 세포의 8구석의 값을 검사하는것을 제외하고는 등값선을 표시하는것과 류사하다. 그림 10-128에서는 격자세포와 등값면과의 사귐점들의 몇가지 실례를 보여 주었다. 등값면은 3각형그물로 모형화되며 마지막형태를 현시하는데 면실감처리알고리듬들이 적용된다.

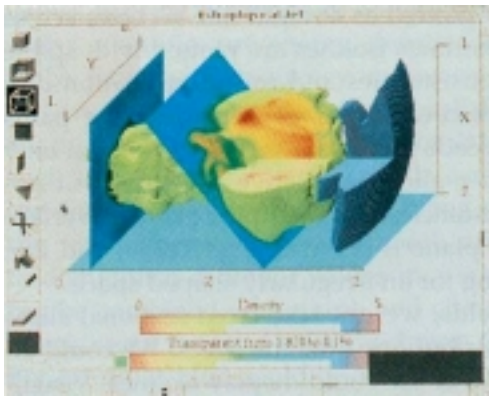



그림 10-126. 3차원자료모임의 자름면

그림 10-127. 우뢰비의 수값모형에서 얻은 물함유량값의 모임으로부터 만든 등값면

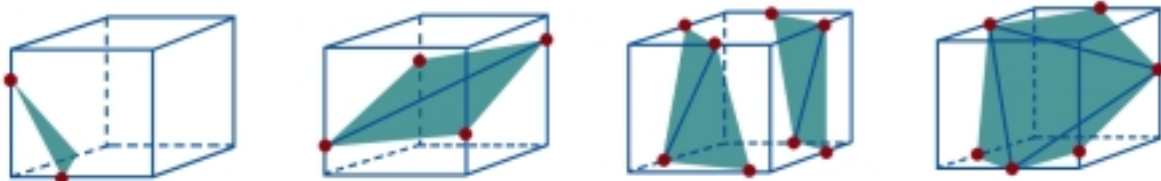



그림 10-128. 격자세포와 등값면과의 사귐점(등값면은 3각형쪼각으로 모형화된다.)

체적실감처리(X선그림과 다소 류사하다.)는 3차원자료모임을 가시화하는 다른 방법이다. 자료모임의 내부정보는 10장 15절에서 도입된 광선투사방법을 리용하여 현시화면에 투영된다. 매개 화면화소로부터의 광선경로를 따라(그림 10-129) 내부자료값을 조사하고 현시하기 위하여 부호화한다. 대체로 격자위치들에서의 자료값은 자료공간의 매개 립체소에 대하여 한개 값이 기억되도록 평균된다. 자료를 현시하기 위하여 어떻게 부호화하는가 하는것은 응용에 관계된다. 실례로 지진자료인 경우 매 광선을 따라 최대 및 최소값을 찾는다. 다음에 구간의 너비에 대한 정보와 최소값을 주기 위하여 값들을 색부호화한다. 의학응용에서 자료값은 조직과 뼈층에 대한 0~1사이 범위의 불투명결수이다. 뼈층은 완전히 불투명하며 한편 조직은 약간 투명(낮은 불투명)하다. 매개 광선을 따라 불투명결수

들은 전체가 1과 같거나 클 때까지 또는 광선이 3차원자료격자의 뒤로 탈퇴할 때까지 축적된다. 축적된 불투명값은 다음에 화소세기준위로 현시되며 그것은 회색계조 또는 색일수 있다. 그림 10-130에서는 개의 심장구조를 표현하는 의학자료모임의 체적가시화를 보여 주었다. 이 체적가시화에서는 매개 화소광선을 따라 최대립체소값까지의 거리의 색부호화된 도면이 현시되였다.

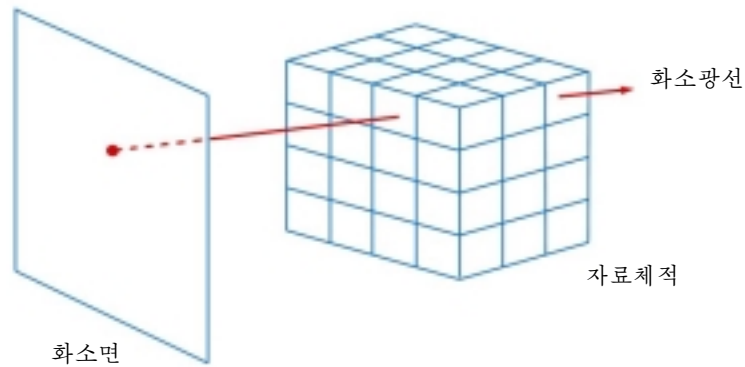



그림 10-129. 내부자료값들을 조사하기 위하여 광선투사를 리용하는 규칙적인 직각자리표자료격자의 체적가시화

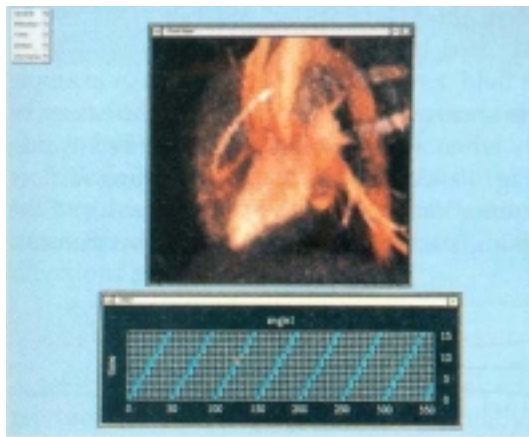



그림 10-130. 매개 화소에 대하여 최대립체소값까지의 색부호화된 거리를 표시하여 얻은 개의 심장에 대한 자료모임의 체적가시화

벡토르마당의 시각적인 표현

3차원공간에서 벡토르량 **V**는 매개 자리표방향에 대하여 하나씩 3개의 스칼라값(V_x, V_y, V_z)을 가지며 2차원벡토르는 두개의 성분(V_x, V_y)을 가진다. 벡토르량을 표현하는 다른 방법은 그의 크기 |**V**|와 방향을 단위벡토르 **u**로 주는것이다. 스칼라와 마찬가지로 벡토르량은 위치, 시간 기타 파라메터들의 함수일수 있다. 물리적인 벡토르량의 몇가지 실례는 속도, 가속도, 힘, 전기마당, 자기마당, 중력마당, 전류이다.

벡토르마당을 가시화하는 한가지 방법은 매 자료점을 벡토르의 크기와 방향을 보여 주는 작은 화살로 표시하는것이다. 이 방법은 그림 10-131에서와 같이 충자름면에 가장 많이 리용된다. 왜냐하면 겹치는 화살들에 의하여 란잡해 진 3차원구역에서 자료의 경향을 보기가 힘들수 있기때문이다. 화살의 길이를 변화시켜 벡토르값의 크기를 보여 줄수 있다. 또는 모든 화살들을 동일한 크기로 만들고 벡토르크기에 대한 선택된 색부호화에 따라 화살들을 서로 다른 색으로 할수 있다.

벡토르값은 또한 마당선 또는 흐름선을 표시하여 표현할수 있다. 마당선은 전기, 자기, 인력마당에 대하여 공통으로 리용된다. 그림 10-132에서 보여 주는바와 같이 벡토르값의 크기는 마당선들사이의 간격에 의하여 지적되며 방향은 마당선의 접선이다. 벡토르마당의 흐름선도면의 실례를 그림 10-133에 보여 주었다. 흐름선은 소용돌이 또는 회리효과가 있을 때 특히 넓은 화살로 표시할수 있다. 이 실례를 그림 10-134에 주었다. 그것은 우뢰비안에서 소용돌이치는 공기흐름무늬를 표시한다. 류체흐름동화에서 벡토르마당의 움직임은 흐름방향을 따라 립자들을 추적하는 방법으로 가시화할수 있다. 흐름선과 립자를 다같이 리용하는 벡토르마당가시화의 실례를 그림 10-135에 보여 주었다.

그림 10-131. 자름면에서 벡토르마당에 대한 화살표현

때때로 단일위치에서 다중량들이 가시화될 때, 방향이 공간의 일부 구역에서 크게 변하지 않을 때 또는 벡토르방향이 덜 흥미 있을 때 벡토르량의 크기만을 표시한다.

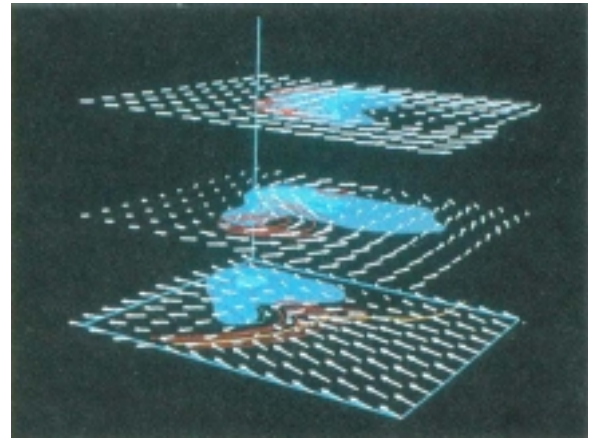
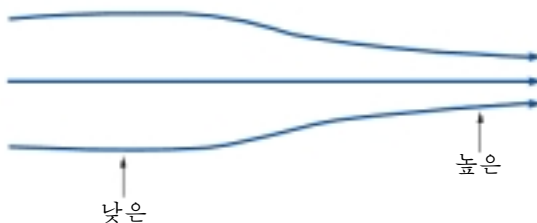



그림 10-132. 벡토르자료모임에 대한 마당선표현

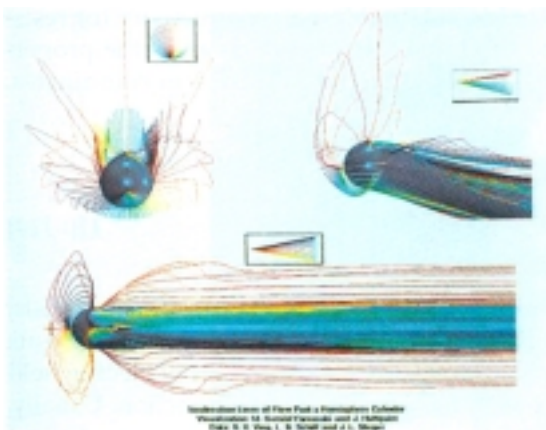



그림 10-133. 기류가 들어 오는 방향에 대하여 약간 기울여 진 반구모자를 가지는 원기둥주위에서 기류를 가시화

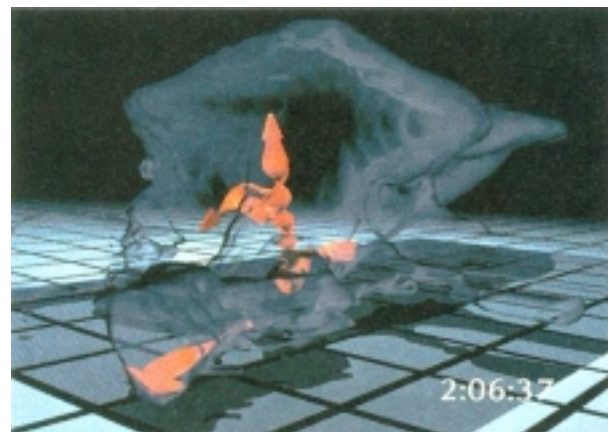



그림 10-134. 꼬이는 기류무늬(우뢰비의 투명한 등값면그림안에서 넓은 흐름선으로 가시화된다.)

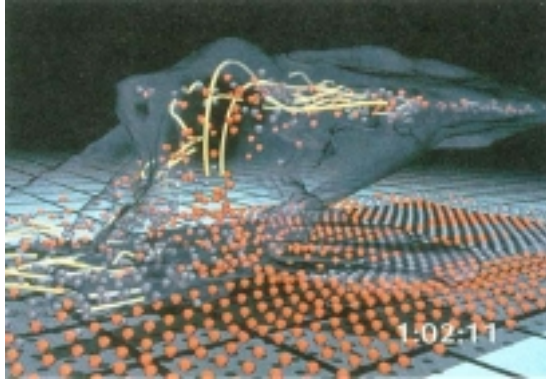



그림 10-135. 우뢰비의 투명한 등값면그림안에서 흐름선과 립자운동에 의하여 가시화되는 기류무늬(오르는 구립자들은 감색이고 떨어 지는 구립자들은 푸른색이다.)

텐소르마당의 시각적인 표현

3차원공간에서 텐소르량은 9개의 성분을 가지며 3×3행렬로 표현할수 있다. 실제로 이 표현은 2차텐소르에 대하여 리용되며 더 높은 차수의 텐소르는 일부 응용들 특히 일반상대성문제에서 나타난다. 물리적인 2차텐소르들의 몇가지 실례는 외부힘을 받는 재료에서 응력과 변형, 전도체의 전도성(또는 저항성), 개개 자리표공간의 특성을 주는 계량텐소르이다. 실레로 직각자리표에서 응력텐소르는

$$\begin{bmatrix} \sigma_x & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_y & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_z \end{bmatrix} \qquad (10-114)$$

와 같이 표현할수 있다.

텐소르량은 서로 다른 방향에서 서로 다른 특성을 가지는 이방성재료에서 자주 만난다. 실레로 전도성텐소르의 x, xy, xz원소는 x, y, z방향의 전기마당성분이 x방향전류에 기여하는 몫을 표현한다. 보통 물리적인 텐소르량들은 대칭이며 따라서 텐소르는 6개의 서로 다른 값들을 가진다. 실레로 응력텐소르의 xy 및 yx성분들은 같다.

2차텐소르량의 6개의 성분들을 모두 표현하는 가시화계획은 6개의 파라메터를 가지는 형태를 창안하는데 기초한다. 텐소르에 대한 한가지 도형적인 표현을 그림 10-136에 보여 준다. 텐소르의 3개의 대각선원소들은 화살의 크기와 방향을 만드는데 리용되며 3개의 비대각선항들은 타원판의 형태와 색갈을 설정하는데 리용된다.

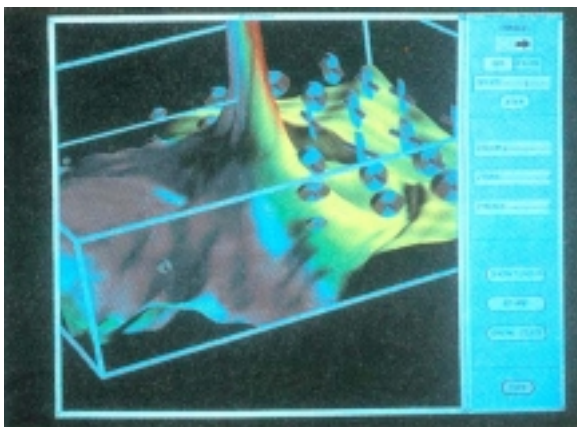



그림 10-136. 응력재료의 걸면에서 타원판과 화살에 의한 응력 및 변형텐소르의 표현

텐소르량의 6개 성분을 모두 가시화하는 대신에 텐소르를 벡토르 또는 스칼라로 줄일수 있다. 벡토르표현을 리용하여 텐소르의 대각선원소들에 대한 벡토르표현을 간단히 현시할수 있다. 그리고 텐소르수축연산을 적용하여 스칼라표현을 얻을수 있다. 실례로 응력 및 변형텐소르는 외부힘을 받는 재료의 점들에서 표시될수 있는 스칼라변형에네르기밀도를 얻기 위하여 수축될수 있다(그림 10-137).

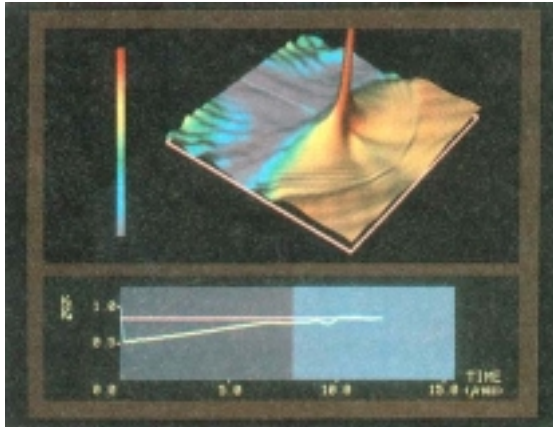



그림 10-137. 응력재료의 결면에서의 균렬전파의 가시화에서 응력 및 변형텐소르를 변형에네르기 밀도그림에 의하여 표현

다변량자료마당의 시각적인 표현

일부 응용들에서 공간의 일부구역의 매 격자위치에서 다변량자료값을 가질수 있으며 그것은 스칼라, 벡토르, 지어 텐소르값들의 혼합일수 있다. 실례로 류체흐름문제에서는 매 3차원위치에서 류체의 속도, 온도, 밀도값들을 가질수 있다. 따라서 매 위치에서 현시하기 위한 5개의 스칼라값을 가지며 이것은 텐소르마당의 현시와 류사하다.

다변량자료마당을 현시하는 방법은 다중부분들에 의하여 도형적인 물체(때때로 글리프(glyphs)라고 한다.)를 만드는것이다. 글리프의 매 부분은 물리적량을 표현한다. 매 부분의 크기와 색은 스칼라크기에 대한 정보를 현시하는데 리용될수 있다. 벡토르마당의 방향적인 정보를 주기 위하여 벡토르를 표현하는 글리프부분으로서는 쐐기, 원추 또는 기타 다른 지적형태를 리용할수 있다. 선택된 격자위치들에서 글리프구조를 리용하는 다변량자료마당의 가시화실례를 그림 10-138에 보여 주었다.

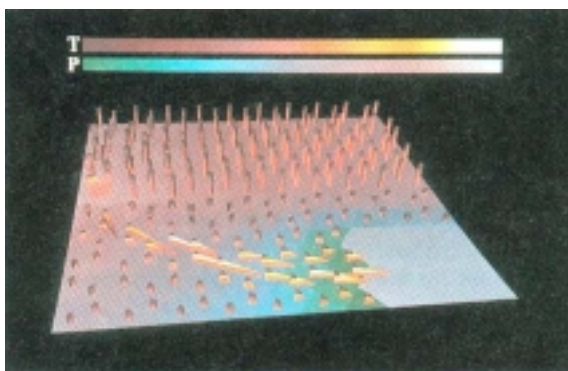



그림 10-138. 글리프를 리용하는 다변량자료마당의 동화된 가시화의 한 프레임(글리프의 쐐기모양부분은 매점에서 벡토르량의 방향을 지적한다.)

요약

도형처리장면에 현시할 다종다양한 물체들을 모형화하기 위한 여러가지 표현방법들이 개발되였다. 《표준도형처리물체》들은 다각형쪼각들의 그물에 의하여 표현된다. 일반적으로 다각형그물표현은 기타 표현들로부터 유도된다.

2차곡면과 같은 곡면함수들은 구와 기타 원활한 곡면들을 표현하는데 리용된다. 설계응용에서는 원활한 곡면형태들을 표현하는 초2차곡면, 스플라인, 무정형물체들을 리용할수 있다. 게다가 CSG와 스위프표현과 같은 구성기법들은 보다 간단한 형태들로부터 복합한 형태의 물체들을 만드는데 쓸모 있다. 그리고 8분나무표현에서는 물체의 걸면뿐아니라 내부정보도 기억시킬수 있다.

나무와 구름과 같은 자연물체들과 기타 불규칙적인 형태의 물체들은 프락탈, 형태문법, 립자계에 의하여 표현할수 있다. 마지막으로 가시화기술은 수값자료모임이나 기타 류형의 자료모임을 현시하기 위하여 도형처리표현들을 리용한다. 여러가지 류형의 수값자료에는 스칼라값, 벡토르값, 텐소르값들이 있다. 또한 많은 과학적가시화에서는 여러가지 자료류형들이 결합된 다변량자료모임을 표현하기 위한 방법들이 요구된다.

참고문헌

초2차곡면의 구체적인 설명은 Barr(1981)에 주었다. 무정형물체모형화에 대한 정보는 Blinn(1982)에서 보시오. 메타볼모형은 Nishimura(1985)에서 설명되며 유연한 물체모형은 Wyville, Wyville, McPheeters(1987)에서 설명된다.

보조변수곡선과 곡면표현의 정보는 Bezier(1972), Burt와 Adelson(1983), Barsky(1983, 1984), Kochanek 와 Bartels(1984), Farouki 와 Hinds(1985), Huitric 와 Nahas(1985), Mortenson(1985), Farin(1988), Rogers와Adams(1990)에 주었다.

8분나무와 4분나무는 Doctor(1981), Yamaguchi, Kunii, Fujimura(1984), Carlbom, Chakravarty, Vanderschel (1985)에서 설명된다. 립체모형화참고서들은 Casale과 Stanton(1985), Requicha과 Rossignac 1992)이다.

프락탈표현에 대한 정보는 Mandelbrot(1977, 1982), Fournier, Fussel, Carpenter(1982), Norton(1982), Peitgen과 Richter(1986), Peitgen과 Saupe (1988), Koh와 Hearn(1992), Barnsley(1993)에서 보시오. 형태문법은 Glassner(1992)에서 설명되며 립자계는 Reeves(1983)에서 설명된다. 물리학적모형화의 설명은 Barzel(1992)에 주었다.

가시화방법들에 대한 일반적인 소개는 Hearn와 Barker(1991)에 주었다. 구체적인 가시화방법들의 추가적인 정보는 Sabin(1985), Lorensen 과 Cline(1987), Drebin, Carpenter, Hanrahan(1988), Sabella(1988), Upson 과 Keeler(1988), Frenkel(1989), Nielson, Shriver, Rosenblum(1990), Nielson (1993)에서 보시오. 정보의 시각적인 현시를 위한 안내는 Tufte(1983, 1990)에 주었다.

련습문제

10-1. 단위바른6면체에 대하여 그림 10-2에서와 같이 기하학적자료표를 설정하시오.

10-2. 정점표와 다각형표(ㄱ), 단일한 다각형표(ㄴ)만을 리용하여 단위바른6면체에 대한 기하학적자료표를 설정하시오. 단위바른6면체를 표현하기 위한 두가지 방법을 세개의 자료표를 리용하는 표현방법과 비교하고 매 방법의 기억기요구를 추정하시오.

10-3. 원기둥에 대한 효률적인 다각형표현을 정의하고 표현의 선택을 조정하시오.

10-4. 물체를 정의하는 자료점들을 임의로 입력할 때 다각형표를 확립하는 절차를 설정하시오.

10-5. 그림 10-2의 자료표에 대하여 일관성과 완전성을 검사하는 루틴을 개발하시오.

10-6. 물체를 정의하는 3차원평면들의 임의의 모임에 대하여 파라메터 A, B, C, D를 계산하는 프로그람을 쓰시오.

10-7. 물체의 모든 면들에 대하여 평면의 파라메터 A, B, C, D가 주어 졌을 때 임의로 지적

된 점이 물체의 내부에 있는가 또는 외부에 있는가를 결정하는 알고리듬을 개발하시오.

10-8. 자리표계가 오른손계로부터 왼손계로 변화될 때 평면의 방정식에서 파라메터 A, B, C, D의 값들은 어떻게 변경되여야 하는가?

10-9. 임의로 지적된 구, 타원체, 원기둥을 다각형그물표현으로 변환하기 위한 알고리듬을 설정하시오.

10-10. 지적된 초타원체를 다각형그물표현으로 변환하기 위한 알고리듬을 설정하시오.

10-11. 메타볼표현을 다각형그물표현으로 변환하기 위한 알고리듬을 설정하시오.

10-12. xy평면에서 조종점들의 입력모임이 주어 졌을 때 2차원의 기본스플라인곡선을 현시하는 루틴을 쓰시오.

10-13. xy평면에서 조정점들의 입력모임이 주어 졌을 때 2차원의 코카니크-바텔스곡선을 현시하는 루틴을 쓰시오.

10-14. 3개의 조종점에 대한 2차베지에혼합함수들을 결정하시오. 매 함수의 곡선을 그리고 최대 및 최소값을 표식하시오

10-15. 5개의 조종점에 대한 베지에혼합함수들을 결정하시오. 매 함수의 곡선을 그리고 최대 및 최소값을 표식하시오.

10-16. xy평면에서 4개의 조종점들이 주어 졌을 때 2차원의 3차베지에곡선을 현시하는 효률적인 루틴을 쓰시오.

10-17. 매개 부분에 대하여 1계련속인 2차원의 3차베지에곡선형태를 설계하는 루틴을 쓰시오. xy평면에서 곡선의 매 부분에 대하여 조종점들의 위치를 선택하는데 대화식방법을 리용하시오.

10-18. 매개 부분에 대하여 2계련속인 2차원의 3차베지에곡선형태를 설계하는 루틴을 쓰시오. xy평면에서 곡선의 매 부분에 대한 조종점들의 위치를 선택하는데 대화식방법을 리용하시오.

10-19. 부분분할방법을 리용하여 3차베지에곡선을 현시하는 루틴을 쓰시오.

10-20. d=5일 때 균일주기B스플라인곡선의 혼합함수들을 결정하시오.

10-21. d=6일 때 균일주기B스플라인곡선의 혼합함수들을 결정하시오.

10-22. 조종점들의 입력모임이 주어 졌을 때 2차원의 균일주기3차B스플라인곡선우의 점들을 계산하기 위하여 앞계차를 리용하는 프로그람을 쓰시오.

10-23. 유리베지에스플라인표현을 리용하여 xy평면에서 임의로 지적된 원추곡선을 현시하는 루틴을 쓰시오.

10-24. 유리B스플라인표현을 리용하여 xy평면에서 임의로 지적된 원추곡선을 현시하는 루틴을 쓰시오.

10-25. 점 $\mathbf{P}(u, v)$에서 베지에곡면의 법선벡토르를 계산하기 위한 알고리듬을 개발하시오.

10-26. 곡선경로우의 점들을 계산할 때 앞계차를 리용하여 임의로 지적된 2차곡선을 현시하는 프로그람을 쓰시오.

10-27. 곡선경로우의 점들을 계산할 때 앞계차를 리용하여 임의로 지적된 3차곡선을 현시하는 프로그람을 쓰시오.

10-28. 임의로 지적된 2차곡선에 대하여 앞계차를 계산하기 위한 식들을 유도하시오.

10-29. 임의로 지적된 3차곡선에 대하여 앞계차를 계산하기 위한 식들을 유도하시오.

10-30. 평행이동스위프에 의하여 물체를 정의하는 입력파라메터들로부터 3차원물체의 표현을 만들기 위한 절차를 개발하시오.

10-31. 회전스위프에 의하여 물체를 정의하는 입력파라메터들을 리용하여 3차원물체의 표현을 만들기 위한 절차를 개발하시오.

10-32. 립체구성기하방법을 리용하여 3차원기초요소들의 결합으로 립체물체를 만들기 위한 알고리듬을 창안하시오. 매 기초요소는 면들의 모임으로 정의된다.

10-33. 8분나무구조로 정의되는 립체들의 기초요소모임을 리용하여 립체구성기하모형화를 수행하기 위한 알고리듬을 개발하시오.

10-34. 2차원장면을 4분나무표현으로 부호화하기 위한 알고리듬을 개발하시오.

10-35. 장면을 현시하기 위하여 장면의 4분나무표현을 프레임완충기에 적재하기 위한 알고리듬을 설정하시오.

10-36. 3차원물체의 다각형정의를 8분나무표현으로 변환하는 루틴을 쓰시오.

10-37. 우연중점변위법을 리용하여 xy평면의 수평선에서 시작하여 산의 륜곽선을 만드는 루틴을 쓰시오.

10-38. 우연중점변위법을 리용하여 바닥면우의 높이를 계산하는 루틴을 쓰시오.

10-39. 반복회수가 임의로 주어 질 때 프락탈눈송이(코흐곡선)를 만들기 위한 프로그람을 쓰시오.

10-40. 그림 10-71 또는 10-72의 발생무늬들중 하나를 리용하며 반복회수가 주어 졌을 때 프락탈곡선을 만드는 프로그람을 쓰시오. 곡선의 프락탈차원은 무엇인가?

10-41. 자기두제곱함수 $f(z)= z^2+\lambda$를 리용하여 프락탈곡선을 만드는 프로그람을 쓰시오. 여기서 λ는 임의로 지적된 복소상수이다.

10-42. 자기두제곱함수 $f(z)=i(z^2+1)$를 리용하여 프락탈곡선을 만드는 프로그람을 쓰시오. 여기서 $i=\sqrt{-1}$ 이다.

10-43. 만델브로트모임을 현시하기 위하여 서로 다른 색결합들을 대화식으로 선택하는 루틴을 쓰시오.

10-44. 만델브로트모임의 임의의 직4각형구역을 대화식으로 선택하고 선택된 구역을 확대하는 프로그람을 쓰시오.

10-45. 임의로 지적된 원과 임의로 주어 진 점에 대하여 식 10-112의 점의 반전을 실현하는 루틴을 쓰시오.

10-46. 바른3각형의 형태를 변경시키기 위한 기하학적인 치환규칙들의 모임을 설계하시오.

10-47. 기하학적인 치환규칙들의 모임이 주어 졌을 때 바른3각형을 다른 형태로 변환하는 단계들을 현시하는 프로그람을 쓰시오.

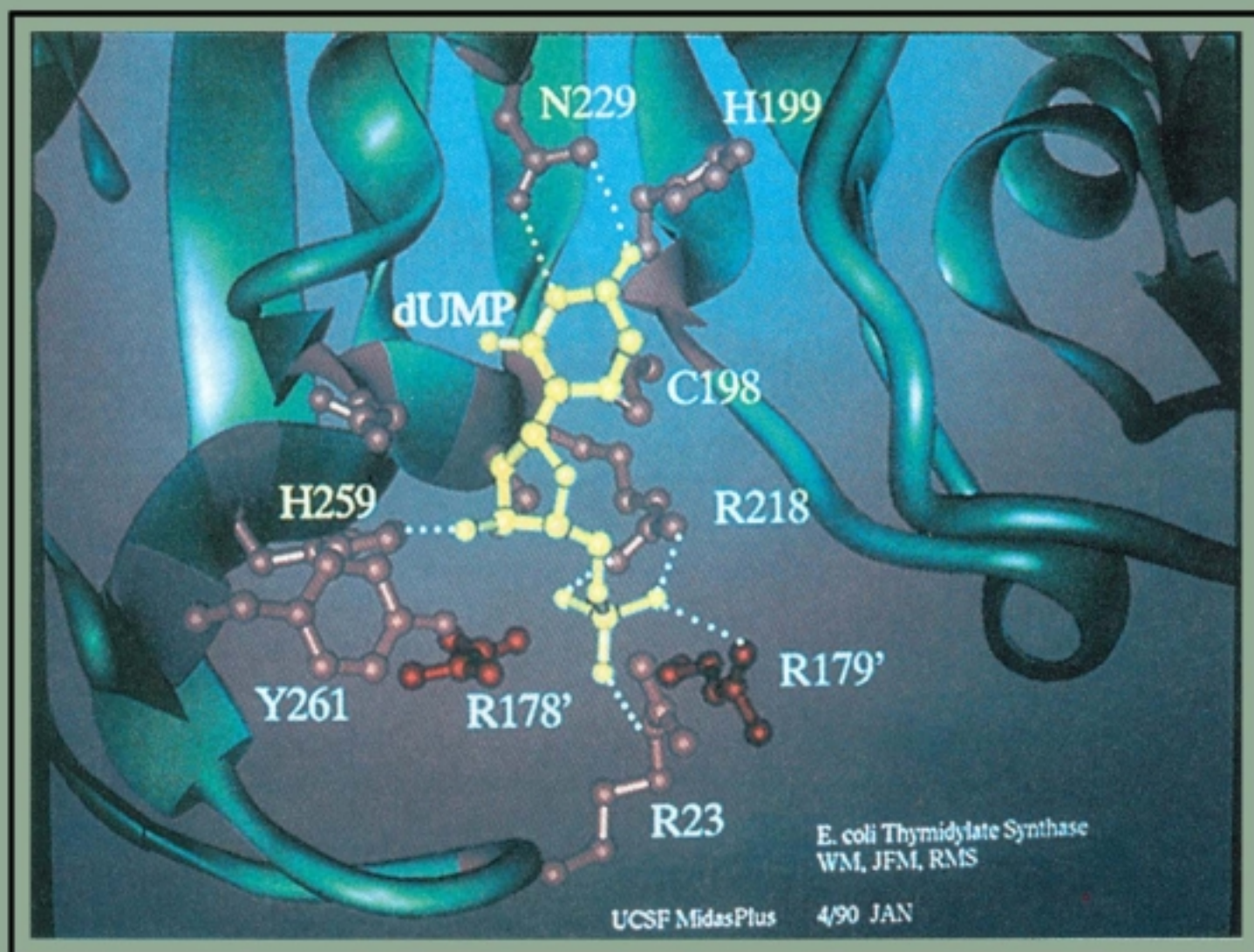
10-48. 립자계를 리용하여 xy평면에서 폭발하는 축포를 모형화하는 프로그람을 쓰시오.

10-49. 직4각형의 네 변에서 꼭같은 용수철들을 리용하여 직4각형을 비강성체로 모형화하기 위한 알고리듬을 개발하시오.

10-50. 허위색방법을 리용하여 2차원의 스칼라자료모임을 가시화하는 루틴을 쓰시오.

10-51. 륜곽선을 리용하여 2차원의 스칼라자료모임을 가시화하는 루틴을 쓰시오.

10-52. 벡토르값에 대하여 화살표현을 리용해서 2차원의 벡토르자료모임을 가시화하는 루틴을 쓰시오. 모든 화살들의 길이는 갈게 하고 벡토르의 서로 다른 크기들을 표현하기 위하여 서로 다른 색갈로 화살들을 현시하시오.

11장. 3 차원기하학적 변환과 모형화변환

3차원기하학적변환방법과 물체모형화방법은 2차원방법에 z자리표를 도입하여 확장한다. 물체가 3개의 자리표방향으로 각각 얼마나 움직이는가를 결정하는 3차원평행이동벡토르를 지적하여 물체를 평행이동시킨다. 류사하게 3개의 자리표비례결수를 리용하여 물체를 비례변환한다. 3차원회전에로의 확장은 그리 간단하지 않다. xy평면에서 2차원회전을 설명할 때에는 xy평면에 수직인 축에 대한 회전만을 고찰하였다. 3차원공간에서는 임의의 방향의 회전축을 선택할수 있다. 대부분의 도형처리프로그람들에서는 3개의 직각자리표축들에 대한 회전을 합성하여 3차원회전을 얻는다. 한편 회전축의 방향과 회전각이 주어 지면 일반회전행렬을 쉽게 만들수 있다. 2차원의 경우와 마찬가지로 기하학적변환들을 행렬형식으로 표현할수 있다. 임의의 순서의 변환렬은 개별적인 변환행렬들을 차례로 련결하여 만든 하나의 행렬로 표현한다.

1절. 평행이동

3차원의 점은 동차자리표를 리용할 때 행렬연산

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{11-1}$$

또는

$$P' = T \cdot P \tag{11-2}$$

에 의해 위치 **P**=(x, y, z)로부터 위치 **P′**=(x', y', z')에로 평행이동된다(그림 11-1). x, y, z 자리표방향에서의 평행이동거리를 지적하는 파라메터 t_x, t_y, t_z에는 임의의 실수값이 주어 진다. 식 11-1의 행렬표현은 3개의 식

$$x' = x + t_x, \qquad y' = y + t_y, \qquad z' = z + t_z, \tag{11-3}$$

과 등가이다.

물체는 물체를 정의하는 매점을 변환하여 3차원적으로 평행이동시킨다. 다각형면들의 모임으로 표현되는 물체인 경우 매면의 정점들을 평행이동시키고 새 위치에서 다각형면들을 다시 그린다(그림 11-2).

평행이동거리 t_x, t_y, t_z를 부로 하면 식 11-1의 역평행이동행렬을 얻는다. 이것은 반대방향의 평행이동을 만들며 평행이동행렬과 그의 역행렬의 적은 단위행렬로 된다.

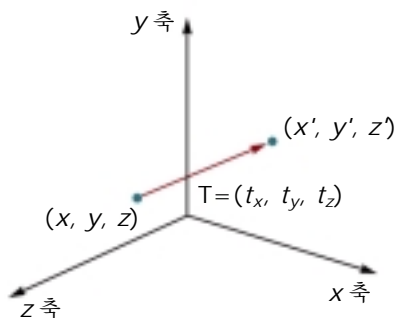



그림 11-1. 평행이동벡토르 T=(t_x, t_y, t_z)에 의한 점의 평행이동

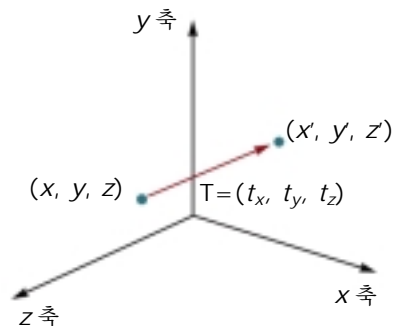



그림 11-2. 평행이동벡토르 T에 의한 물체의 평행이동

2절. 회전

물체의 회전변환을 얻으려면 물체가 회전하는 회전축과 회전각을 지적하여야 한다. 모든 변환들이 xy평면에서 수행되는 2차원회전과 달리 3차원회전은 공간상의 임의의 축에 대하여 진행될수 있다. 처리하기 제일 쉬운 회전축은 자리표축에 평행인 축들이다. 그리고 임의의 일반회전은 자리표축들에 대한 회전들을 결합하여(적당한 평행이동과 함께) 진행할수 있다.

일반적으로 정의 회전각은 정의 자리표축에서 자리표원점을 볼 때 자리표축에 대한 시계바늘반대방향으로 회전시키는 각으로 정의한다(그림 11-3). 이것은 xy평면에서 정의 회전을 z축에 평행인 축에 대한 시계바늘반대방향의 회전으로 정의한것과 일치한다.

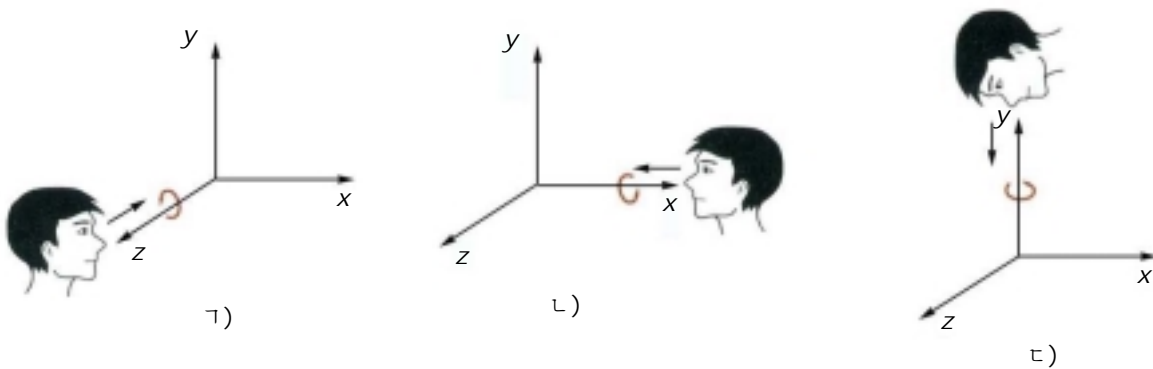



그림 11-3. 자리표축에 대한 정의 회전방향은 정의 자리표축위치에서 원점을 볼 때 시계바늘반대방향으로 한다.

자리표축에 대한 회전

2차원의 z축에 대한 회전식은 3차원에로 쉽게 확장할수 있다.

$$\begin{aligned} x' &= x\cos\theta - y\sin\theta \\ y' &= x\sin\theta + y\cos\theta \\ z' &= z \end{aligned} \qquad (11-4)$$

파라메터 θ는 회전각을 표시한다. 3차원의 ㄱz축에 대한 회전식은 동차자리표형식에서

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (11-5)$$

로 표시되며

$$\mathbf{P}' = \mathbf{R}_z(\theta) \cdot \mathbf{P} \qquad (11-6)$$

와 같이 더 간단하게 쓸수 있다. 그림 11-4에서는 z축에 대한 물체의 회전을 보여 주었다.

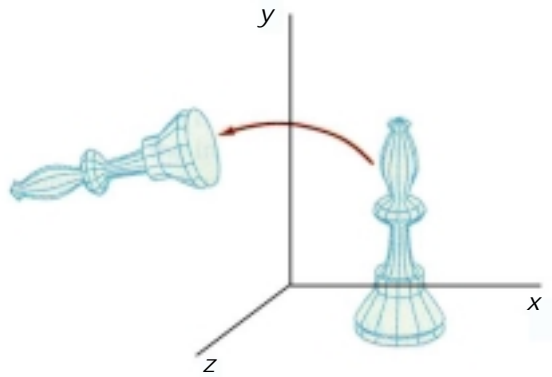



그림 11-4. z축에 대한 물체의 회전

다른 두 자리표축에 대한 회전변환식은 식 11-4에서 자리표파라메터 x, y, z를 순환적으로 바꾸어 얻을수 있다. 즉 그림 11-5에서 보여 주는바와 같이 교환

$$x \to y \to z \to x \qquad (11-7)$$

를 리용한다.

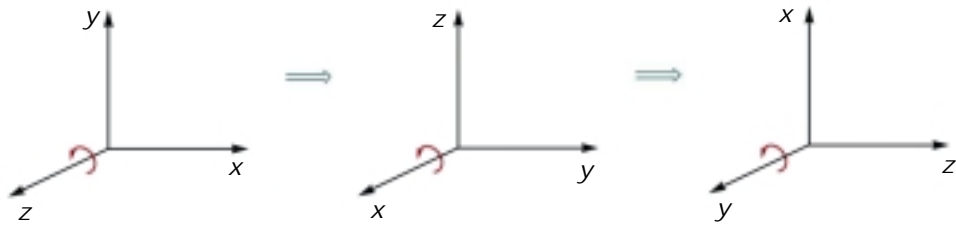



그림 11-5. 3개의 자리표축에 대한 회전식을 만들기 위한 직각자리표축들의 순환적인 교환

식 11-4에 식 11-7을 대입하면 x축에 대한 회전식을 얻는다.

$$\begin{aligned} y' &= y\cos\theta - z\sin\theta \\ z' &= y\sin\theta + z\cos\theta \\ x' &= x \end{aligned} \qquad (11-8)$$

이것은 동차자리표형식

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (11-9)$$

또는

$$\mathbf{P}' = \mathbf{R}_x(\theta) \cdot \mathbf{P} \qquad (11-10)$$

로 쓸수 있다. x축에 대한 물체의 회전을 그림 11-6에 보여 주었다.

식 11-8에서 자리표들을 순환적으로 바꾸면 y축회전에 대한 변환식을 얻는다.

$$z' = z\cos\theta - x\sin\theta$$
$$x' = z\sin\theta + x\cos\theta \qquad (11\text{-}11)$$
$$y' = y$$

y축회전에 대한 행렬표현은

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (11\text{-}12)$$

또는

$$\mathbf{P}' = \mathbf{R}_y(\theta) \cdot \mathbf{P} \qquad (11\text{-}13)$$

이다. y축에 대한 회전의 실례를 그림 11-7에 보여 주었다.

역회전행렬은 회전각 θ를 $-\theta$로 교체하여 얻는다. 부의 회전각은 시계바늘방향의 회전을 만들므로 임의의 회전행렬에 그의 역회전행렬을 곱하면 단위행렬이 된다. 회전각의 부호변화는 시누스함수에만 영향을 미치기때문에 역회전행렬은 행과 렬을 서로 교환하여서도 얻을수 있다. 즉 임의의 회전행렬 $\mathbf{R}$의 역회전행렬은 전위행렬을 계산하여 얻을수 있다($\mathbf{R}^1 = \mathbf{R}^T$). 역회전행렬을 얻는 이 방법은 임의의 합성회전행렬에서도 성립한다.

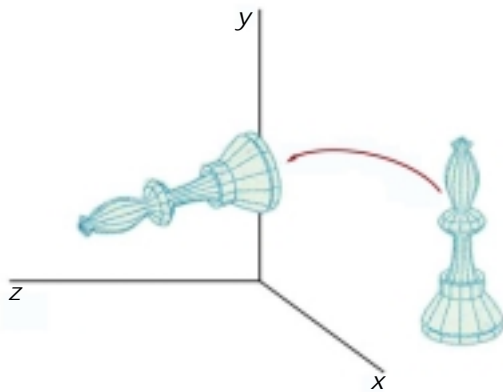



그림 11-6. x축에 대한 물체의 회전

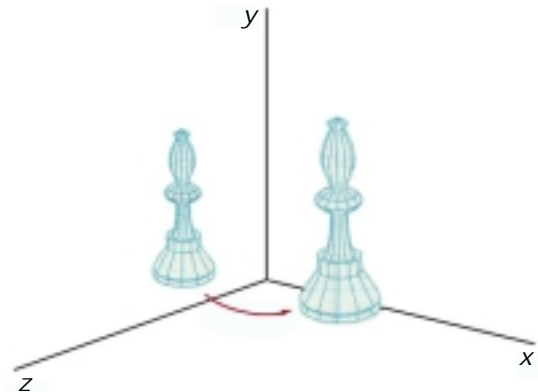



그림 11-7. y축에 대한 물체의 회전

일반적인 3차원회전

자리표축과 일치하지 않는 임의의 축에 대한 회전행렬은 평행이동과 자리표축에 대한 회전을 결합하는 합성변환으로 얻을수 있다. 먼저 주어 진 회전축을 어느 한 자리표축에로 이동시키는 변환렬을 만들고 다음 그 자리표축에 대하여 지적된 회전각으로 회전시키고 마지막에 회전축을 자기 본래 위치로 돌려 보내는 역변환렬을 만든다.

물체를 어느 한 자리표축에 평행인 축에 대하여 회전시키는 특수한 경우에는 다음의 변환렬로 요구되는 회전을 얻을수 있다.

1. 회전축을 평행자리표축과 일치하도록 물체를 평행이동시킨다.
2. 그 축에 대하여 지적된 회전을 수행한다.

3. 회전축이 자기 본래위치로 되돌아 가도록 물체를 평행이동시킨다.

이 변환렬의 걸음들을 그림 11-8에서 보여 주었다. 이 그림에서 물체의 임의의 자리표위치 **P**는

$$\mathbf{P}' = \mathbf{T}^{-1} \cdot \mathbf{R}_x(\theta) \cdot \mathbf{T} \cdot \mathbf{P}$$

와 같이 변환된다. 합성변환행렬은

$$\mathbf{R}(\theta) = \mathbf{T}^{-1} \cdot \mathbf{R}_x(\theta) \cdot \mathbf{T}$$

이며 이것은 2차원에서 임의의 점주위로의 회전행렬과 같은 형식이다.

어느 자리표축에도 평행이 아닌 축에 대하여 물체를 회전시킬 때에는 추가적인 변환들을 수행하여야 한다. 이 경우에 회전축을 선택된 자리표축과 일치시키는 회전과 회전축을 자기 본래방향으로 돌려 보내는 회전이 요구된다. 회전축과 회전각이 주어 지면 요구되는 회전을 다음의 다섯걸음으로 수행할수 있다.

1. 회전축이 자리표원점을 통과하도록 물체를 평행이동시킨다.
2. 회전축이 한 자리표축과 일치하도록 물체를 회전시킨다.
3. 그 자리표축에 대하여 지적된 회전을 수행한다.
4. 회전축을 자기 본래방향으로 돌려 보내기 위하여 역회전을 적용한다.
5. 회전축을 자기 본래위치로 돌려 보내기 위하여 역평행이동을 적용한다.

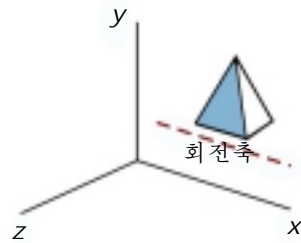
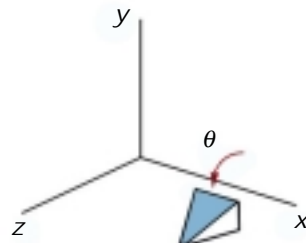
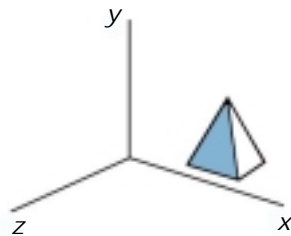
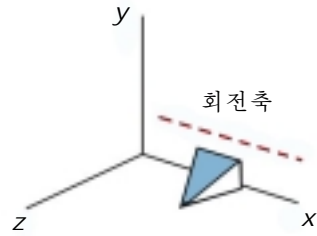



그림 11-8. x축에 평행인 축에 대하여 물체를 회전시키는 변환렬

회전축은 3개의 자리표축가운데서 임의의 축으로 잡을수 있다. z축을 선택하는것이 좋다. 그림 11-9에서는 회전축을 z축과 일치시키고 다시 자기 본래위치로 돌려 보내기 위한 변환행렬들의 설정방법을 보여 주었다.

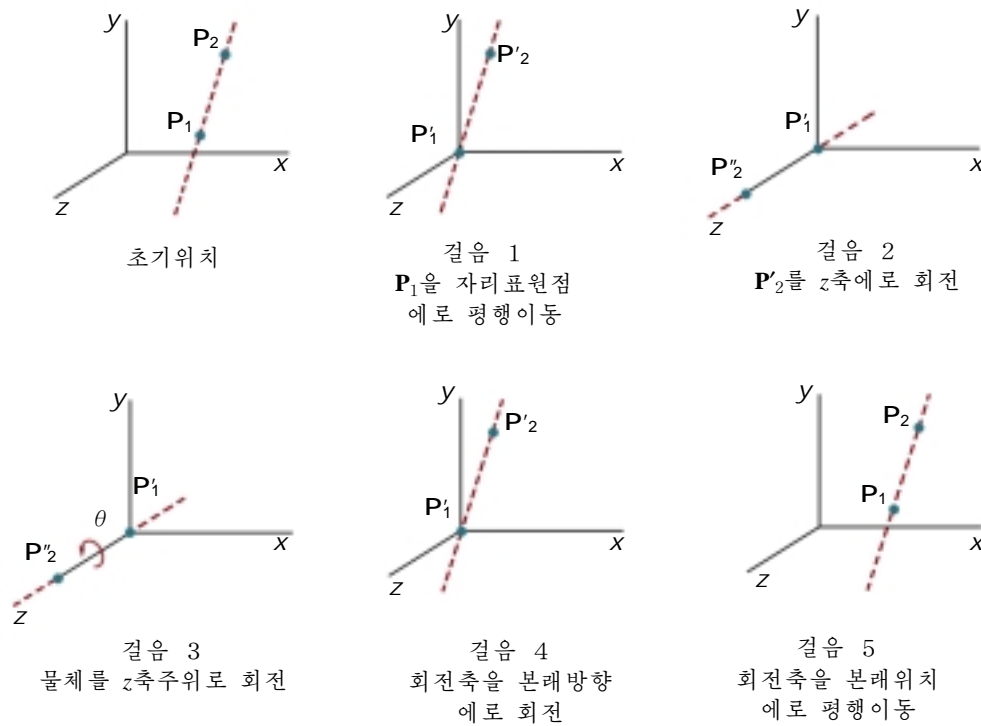



그림 11-9. 임의의 축에 대한 회전행렬을 얻는 5개의 변환걸음
(회전축은 z축에 투영된다.)

회전축은 그림 11-10에서와 같이 두개의 자리표위치에 의하여 정의할수도 있고 또는 한개의 자리표점과 회전축과 두 자리표축사이의 방향각(또는 방향코시누스)에 의하여 정의할수도 있다. 회전축을 두 점에 의하여 정의하고 회전방향은 $\mathbf{P}_2$에서 $\mathbf{P}_1$을 볼 때 회전축에 대한 시계바늘반대방향이라고 가정하자. 그러면 회전축벡토르는 두 점에 의하여

$$\begin{aligned}\mathbf{V} &= \mathbf{P}_2 - \mathbf{P}_1 \\ &= (x_2 - x_1,\ y_2 - y_1,\ z_2 - z_1)\end{aligned} \qquad (11-14)$$

와 같이 정의된다. 단위회전축벡토르 $\mathbf{u}$는

$$\mathbf{u} = \frac{\mathbf{V}}{|\mathbf{V}|} = (a, b, c) \qquad (11-15)$$

와 같이 정의된다. 여기서 단위벡토르 $\mathbf{u}$의 성분 a, b, c는 회전축의 방향코시누스이다.

$$a = \frac{x_2 - x_1}{|\mathbf{V}|},\ b = \frac{y_2 - y_1}{|\mathbf{V}|},\ c = \frac{z_2 - z_1}{|\mathbf{V}|} \qquad (11-16)$$

회전이 반대방향($\mathbf{P}_2$에서 $\mathbf{P}_1$을 볼 때 시계바늘방향)이면 $\mathbf{P}_2$에서 $\mathbf{P}_1$을 가리키도록 회전축벡토르 $\mathbf{V}$와 단위벡토르 $\mathbf{u}$를 반전시킨다.

요구하는 회전에 대한 변환렬에서 첫 걸음은 회전축이 자리표원점을 통과하도록 위치를 옮기는 평행이동행렬을 설정하는것이다. 이것은 지적된 회전방향(그림 11-10)에 대하여 점 $\mathbf{P}_1$을 자리표원점에로 움직여서 수행한다(회전방향이 반대방향으로 지적되였을 때에는 $\mathbf{P}_2$를 자리표원점에로 움직인다.). 이 평행이동행렬은

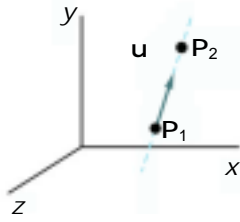



그림 11-10. 점 $\mathbf{P}_1$과 $\mathbf{P}_2$에 의하여 정의되는 회전축(파선)(단위회전축 벡토르 **u**의 방향은 지적된 회전 방향에 의하여 결정된다.)

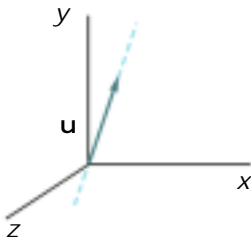



그림 11-11. 회전축을 자리표 원점에로 평행이동

$$\mathbf{T}=\begin{bmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11-17}$$

이다. 이것은 그림 11-11에서와 같이 회전축과 물체의 위치를 옮긴다.

이제는 회전축을 z축과 일치시켜야 한다. 이것을 자리표축회전을 리용하여 두 걸음으로 수행할수 있는데 여기에는 몇가지 방법이 있다. 먼저 벡토르 **u**를 xz평면에 넘기기 위하여 x축주위로 회전시키고 다음 y축주위로 회전시켜 **u**를 z축에 일치시킨다. 벡토르 **u**의 이 두 회전을 그림 11-12에 보여 주었다.

회전행렬은 시누스 및 코시누스함수로 되여 있으므로 두 회전행렬의 원소들을 얻는데 표준벡토르연산(부록 2)을 리용할수 있다. 스칼라적연산은 코시누스항을, 벡토르적연산은 시누스항을 결정할수 있게 한다.

u를 xz평면에 넘기는데 필요한 회전각의 시누스 및 코시누스값을 결정하여 x축회전행렬을 설정한다. 이 회전각은 yz평면에 대한 **u**의 투영과 정의 z축사이의 각이다(그림 11-13). yz평면에 대한 **u**의 투영을 벡토르 $\mathbf{u}' = (0, b, c)$로 지적하면 회전각의 코시누스는 $\mathbf{u}'$와 단위z축벡토르 $\mathbf{u}_z$의 스칼라적으로부터 결정할수 있다.

$$\cos\alpha=\frac{\mathbf{u}'\cdot\mathbf{u}_z}{|\mathbf{u}'|\,|\mathbf{u}_z|}=\frac{c}{d} \tag{11-18}$$

여기서 d는 $\mathbf{u}'$의 크기이다.

$$d=\sqrt{b^2+c^2} \tag{11-19}$$

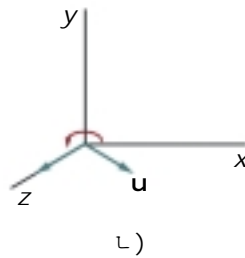
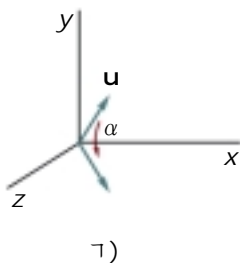



그림 11-12. 단위벡토르 **u**를 xz평면에 넘기기 위하여 x축주위로 회전시키고(ㄱ) 다음 z축과 일치시키기 위하여 y축주위로 회전시킨다(ㄴ).

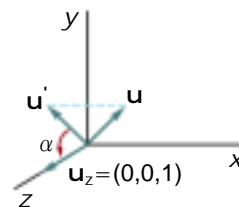



그림 11-13. **u**를 x축주위로 회전시켜 xz평면에 넘기는것은 $\mathbf{u}'$(yz평면에서 **u**의 투영)를 z축에로 각 α만큼 회전시키는 방법으로 한다.

류사하게 각 α의 시누스는 $\mathbf{u}'$와 $\mathbf{u}_z$의 벡토르적으로부터 결정할수 있다. 벡토르적의 자리표독립형식은

$$\mathbf{u}'\times\mathbf{u}_z=\mathbf{u}_x|\mathbf{u}'|\,|\mathbf{u}_z|\,\sin\alpha \tag{11-20}$$

이며 벡토르적의 직각자리표형식은

$$\mathbf{u}' \times \mathbf{u}_z = \mathbf{u}_x \cdot b \qquad (11-21)$$

이다. 식 11-20과 11-21의 오른변을 갈게 하고 $|\mathbf{u}_z|=1$, $|\mathbf{u}'|=d$에 주의하면

$$d\sin\alpha = b$$

또는

$$\sin\alpha = \frac{b}{d} \qquad (11-22)$$

를 얻는다.

벡토르 $\mathbf{u}$의 성분들에 의해 $\cos\alpha$, $\sin\alpha$값들을 결정하였으므로 $\mathbf{u}$의 x축회전행렬을 설정할수 있다.

$$\mathbf{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11-23)$$

이 행렬은 단위벡토르 $\mathbf{u}$를 x축주위로 회전시켜 xz평면에 넘긴다.

다음 xz평면의 단위벡토르를 시계바늘반대방향으로 y축주위로 회전시켜 정의 z축과 일치시키는 변환행렬을 결정하여야 한다. x축주위로 회전시킨후 xz평면에서 단위벡토르의 방향을 그림 11-14에 보여 주었다. $\mathbf{u}''$로 표식하는 이 벡토르의 x성분은 a이다. 왜냐하면 x축회전은 x성분을 변화시키지 않기때문이다. 벡토르 $\mathbf{u}'$는 z축에로 회전되였으므로 $\mathbf{u}''$의 z성분은 d($\mathbf{u}'$의 크기)이다. $\mathbf{u}''$는 xz평면에 있으므로 $\mathbf{u}''$의 y성분은 0이다. 회전각 β의 코시누스는 단위벡토르 $\mathbf{u}''$와 $\mathbf{u}_z$의 스칼라적으로부터 결정할수 있다.

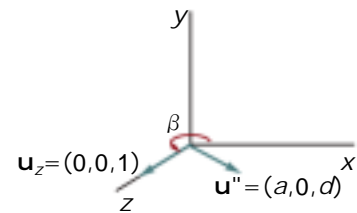



그림 11-14. 단위벡토르 $\mathbf{u}''$($\mathbf{u}$를 xz평면에로 회전한후의 벡토르)의 y축회전(정의 회전각 β는 벡토르 $\mathbf{u}''$를 벡토르 $\mathbf{u}_z$와 일치시킨다.)

$$\cos\beta = \frac{\mathbf{u}'' \cdot \mathbf{u}_z}{|\mathbf{u}''||\mathbf{u}_z|} = d \qquad (11-24)$$

왜냐하면 $|\mathbf{u}_z| = |\mathbf{u}''| = 1$이기때문이다. 벡토르적의 자리표독립형식

$$\mathbf{u}'' \times \mathbf{u}_z = \mathbf{u}_y |\mathbf{u}''||\mathbf{u}_z| \sin\beta \qquad (11-25)$$

와 직각자리표형식

$$\mathbf{u}'' \times \mathbf{u}_z = \mathbf{u}_y \cdot (-a) \qquad (11-26)$$

를 비교하면

$$\sin\beta = -a \qquad (11-27)$$

를 구한다. 이리하여 $\mathbf{u}''$의 y축회전행렬은

$$\mathbf{R}_y(\beta) = \begin{bmatrix} d & 0 & -a & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11-28)$$

이다.

식 11-17, 11-23, 11-28에 의하여 회전축을 정의 z축과 일치시켰다. 지적된 회전각 θ는 z축회전에 적용할수 있다.

$$\mathbf{R}_z(\theta)=\begin{bmatrix}\cos\theta & -\sin\theta & 0 & 0\\ \sin\theta & \cos\theta & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{11-29}$$

주어 진 축에 대한 회전을 얻자면 회전축을 자기 본래위치로 돌려 보내야 한다. 이것은 식 11-17, 11-23, 11-28의 역변환을 적용하면 된다. 임의의 축에 대한 회전변환행렬은 이 7개의 개별적인 변환들의 합성으로 표현할수 있다.

$$\mathbf{R}(\theta)=\mathbf{T}^{-1}\cdot\mathbf{R}_x^{-1}(\alpha)\cdot\mathbf{R}_y^{-1}(\beta)\cdot\mathbf{R}_z(\theta)\cdot\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\alpha)\cdot\mathbf{T} \tag{11-30}$$

합성회전행렬 $\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\alpha)$를 얻는데서 직관성은 좀 적지만 빠른 방법은 임의의 순서의 3차원회전렬에 대한 합성행렬형식의 우점을 살리는것이다.

$$\mathbf{R}=\begin{bmatrix}r_{11} & r_{12} & r_{13} & 0\\ r_{21} & r_{22} & r_{23} & 0\\ r_{31} & r_{32} & r_{33} & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{11-31}$$

이 행렬의 왼쪽 웃구석 3×3부분행렬은 직교한다. 이것은 이 부분행렬의 행(또는 렬)들이 행렬 **R**에 의하여 각각 x, y, z축에로 회전되는 직교단위벡토르들의 모임을 만든다는것을 의미한다.

$$\mathbf{R}\cdot\begin{bmatrix}r_{11}\\ r_{12}\\ r_{13}\\ 1\end{bmatrix}=\begin{bmatrix}1\\0\\0\\1\end{bmatrix},\quad \mathbf{R}\cdot\begin{bmatrix}r_{21}\\ r_{22}\\ r_{23}\\ 1\end{bmatrix}=\begin{bmatrix}0\\1\\0\\1\end{bmatrix},\quad \mathbf{R}\cdot\begin{bmatrix}r_{31}\\ r_{32}\\ r_{33}\\ 1\end{bmatrix}=\begin{bmatrix}0\\0\\1\\1\end{bmatrix} \tag{11-32}$$

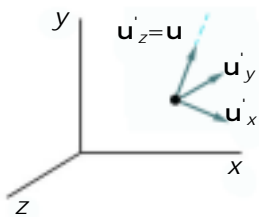



그림 11-15. 단위벡토르 **u**에 의하여 정의되는 회전축에 대한 국부자리표계

따라서 회전축에 의하여 정의되는 국부자리표계를 고찰할수 있으며 행들이 단위국부자리표축벡토르인 행렬을 간단히 만들수 있다. 회전축이 어느 자리표축에도 평행이 아니라고 하면 다음의 단위국부자리표축벡토르모임을 만들수 있다(그림 11-15).

$$\begin{aligned}\mathbf{u}'_z&=\mathbf{u}\\ \mathbf{u}'_y&=\frac{\mathbf{u}\times\mathbf{u}_x}{|\mathbf{u}\times\mathbf{u}_x|}\\ \mathbf{u}'_x&=\mathbf{u}'_y\times\mathbf{u}'_z\end{aligned} \tag{11-33}$$

회전축에 대한 단위국부자리표축벡토르들의 원소들을

$$\begin{aligned}\mathbf{u}'_x&=(u'_{x1},\ u'_{x2},\ u'_{x3})\\ \mathbf{u}'_y&=(u'_{y1},\ u'_{y2},\ u'_{y3})\\ \mathbf{u}'_z&=(u'_{z1},\ u'_{z2},\ u'_{z3})\end{aligned} \tag{11-34}$$

와 같이 표시하면 합성행렬은

$$\mathbf{R}=\begin{bmatrix}u'_{x1} & u'_{x2} & u'_{x3} & 0\\ u'_{y1} & u'_{y2} & u'_{y3} & 0\\ u'_{z1} & u'_{z2} & u'_{z3} & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{11-35}$$

이며 이것은 적 $\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\alpha)$와 같다. 이 행렬은 단위벡토르 $\mathbf{u}'_x$, $\mathbf{u}'_y$, $\mathbf{u}'_z$를 각각 x, y, z축에로 변환한다. $\mathbf{u}'_z=\mathbf{u}$이기때문에 회전축은 z축과 일치된다.

4 원수에 의한 회전

지적된 축에 대한 회전을 얻는 보다 효과적인 방법은 회전변환에 대한 4원수표현을 리용하는것이다. 10장에서는 자기두제곱절차를 리용하여 3차원프락탈을 발생시키는데 4원수를 쓰는 방법을 설명하였다. 4원수는 3차원회전계산을 비롯한 여러가지 콤퓨터도형처리절차들에서 쓸모 있다. 4원수는 4×4행렬보다 적은 기억공간을 요구하며 변환렬에 4원수절차를 쓰면 변환이 더 간단해 진다. 이것은 복잡한 운동렬과 물체의 주어 진 두 위치사이의 운동보간을 요구하는 동화에서 특히 중요하다.

4원수(부록 6)를 특징 짓는 한가지 방법은 스칼라부와 벡토르부로 이루어 지는 순서쌍에 의한 방법이다.

$$q=(s,\ \mathbf{v})$$

4원수는 또한 하나의 실수부(스칼라부)와 3개의 허수부(벡토르 $\mathbf{v}$의 원소들)를 가지는 더 높은 차수의 복소수라고 생각할수 있다. 자리표원점을 통과하는 임의의 축에 대한 회전을 수행하기 위하여 먼저 다음의 스칼라부와 벡토르부를 가지는 단위4원수를 설정한다.

$$s=\cos\frac{\theta}{2},\quad \mathbf{v}=\mathbf{u}\sin\frac{\theta}{2} \qquad (11-36)$$

여기서 $\mathbf{u}$는 선택된 회전축의 단위벡토르이고 θ는 이 축에 대하여 지적된 회전각이다(그림 11-16). 이 4원수에 의하여 회전되는 임의의 점위치 $\mathbf{P}$는 4원수표기로

$$\mathbf{P}=(0,\mathbf{p})$$

와 같이 표현할수 있다. 여기서 점의 자리표는 벡토르부 $\mathbf{p}=(x, y, z)$이다. 점의 회전은 다음에 4원수연산

$$\mathbf{P}'=q\mathbf{P}q^{-1} \qquad (11-37)$$

에 의하여 수행된다. 여기서 $q^{-1}=(s,-\mathbf{v})$는 식 11-36에서 주어 진 스칼라부와 벡토르부를 가지는 단위4원수 q의 역수이다. 이 변환은 스칼라부가 0인 새로운 4원수를 만든다.

$$\mathbf{P}'=(0,\ \mathbf{p}') \qquad (11-38)$$

그리고 벡토르부는 스칼라적과 벡토르적에 의하여

$$\mathbf{p}'=s^2\mathbf{p}+\mathbf{v}(\mathbf{p}\cdot\mathbf{v})+2s(\mathbf{v}\times\mathbf{p})+\mathbf{v}\times(\mathbf{v}\times\mathbf{p}) \qquad (11-39)$$

와 같이 계산된다. 파라메터 s와 $\mathbf{v}$는 식 11-36에서 주어 진 회전값들을 가진다. 3차원물체를 빨리 회전시키기 위하여 대부분의 콤퓨터도형처리체계들에서는 이 벡토르계산을 하드웨어적으로 능률적으로 실현한다.

그림 11-16. 지적된 축주위로의 회전에 대한 단위4원수파라메터 $\mathbf{u}$와 θ

변환식 11-37은 자리표원점을 통과하는 축에 대한 회전과 등가이다. 이것은 회전축을 z축과 일치시키고 z축에 대하여 회전한후 회전축을 자기 본래위치로 돌려 보내는 식 11-30의 회전변환렬과 같다.

부록 6에 주어 진 4원수곱하기의 정의를 리용하고 q의 벡토르부의 성분들을 $\mathbf{v}=(a,\ b,\ c)$로 지적하면 3×3합성회전행렬 $\mathbf{R}_x^{-1}(\alpha)\cdot\mathbf{R}_y^{-1}(\beta)\cdot\mathbf{R}_z(\theta)\cdot\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\alpha)$의 원소들을 얻기 위하여 식 11-39의 항들을

$$\mathbf{M}_R(\theta)=\begin{bmatrix}1-2b^2-2c^2 & 2ab-2sc & 2ac+2sb\\ 2ab+2sc & 1-2a^2-2c^2 & 2bc-2sa\\ 2ac-2sb & 2bc+2sa & 1-2a^2-2b^2\end{bmatrix} \qquad (11-40)$$

과 같이 계산할수 있다. 완전한 일반회전식 11-30을 얻기 위하여서는 회전축을 자리표축에로 움직이고 다시 자기 본래위치로 돌려 보내는 평행이동을 진행하여야 한다. 즉

$$\mathbf{R}(\theta)=\mathbf{T}^{-1}\cdot\mathbf{M}_R\cdot\mathbf{T} \qquad (11-41)$$

실레로 z축회전은 단위4원수의 파라메터들을

$$s=\cos\frac{\theta}{2}, \qquad \mathbf{v}=(0,\ 0,\ 1)\sin\frac{\theta}{2}$$

로 설정하여 수행할수 있다. 여기서 4원수의 벡토르부의 원소들은 $a=b=0$, $c=\sin(\theta/2)$이다. 이 값들을 행렬 11-40에 대입하고 다음의 삼각항등식

$$\cos^2\frac{\theta}{2}-\sin^2\frac{\theta}{2}=1-2\sin^2\frac{\theta}{2}=\cos\theta, \qquad 2\cos\frac{\theta}{2}\sin\frac{\theta}{2}=\sin\theta$$

을 리용하면 변환식 11-5의 3×3 z축회전행렬 $\mathbf{R}_z(\theta)$를 얻는다. 같은 방법으로 단위4원수의 회전값들을 변환식 11-37에 대입하면 식 11-4의 회전된 자리표값을 얻는다.

3절. 비례변환

자리표원점에 대한 위치 $\mathbf{P}=(x, y, z)$의 비례변환의 행렬표현은

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}=\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (11-42)$$

또는

$$\mathbf{P}'=\mathbf{S}\cdot\mathbf{P} \qquad (11-43)$$

와 같이 쓸수 있다. 여기서 비례결수 s_x, s_y, s_z에는 임의의 정의 값이 할당된다. 자리표원점에 대한 비례변환의 양함수적표현은

$$x'=x\cdot s_x, \qquad y'=y\cdot s_y, \qquad z'=z\cdot s_z \qquad (11-44)$$

이다.

변환식 11-42에 의한 물체의 비례변환은 물체의 크기를 변화시키며 자리표원점에 대하여 물체의 위치를 옮긴다. 또한 비례결수들이 모두 같지 않으면 물체의 상대적인 크기는 변화된다. 등방비례변환($s_x=s_y=s_z$)에 의해 물체의 본래형태를 보존한다. 매 비례결수가 꼭같이 2로 설정되는 물체의 비례변환결과를 그림 11-17에 보여 주었다.

그림 11-17. 변환식 11-42에 의하여 물체의 크기를 2배하면 물체는 자리표원점으로부터 더 멀리 움직인다.

선택된 고정점(x_f, y_f, z_f)에 대한 비례변환은 다음의 변환렬로 표현할수 있다.

1. 고정점을 자리표원점에로 평행이동시킨다.
2. 식 11-42를 리용하여 물체를 자리표원점에 대하여 비례변환한다.
3. 고정점을 자기 본래위치에로 역평행이동시킨다.

이 변환렬을 그림 11-18에 보여 주었다. 임의의 고정점에 대한 비례변환의 행렬표현은 평행이동-비례변환-평행이동변환들의 련결로

$$\mathbf{T}(x_f, y_f, z_f)\cdot\mathbf{S}(s_x, s_y, s_z)\cdot\mathbf{T}(-x_f, -y_f, -z_f) = \begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11-45)$$

와 같이 표현할수 있다.

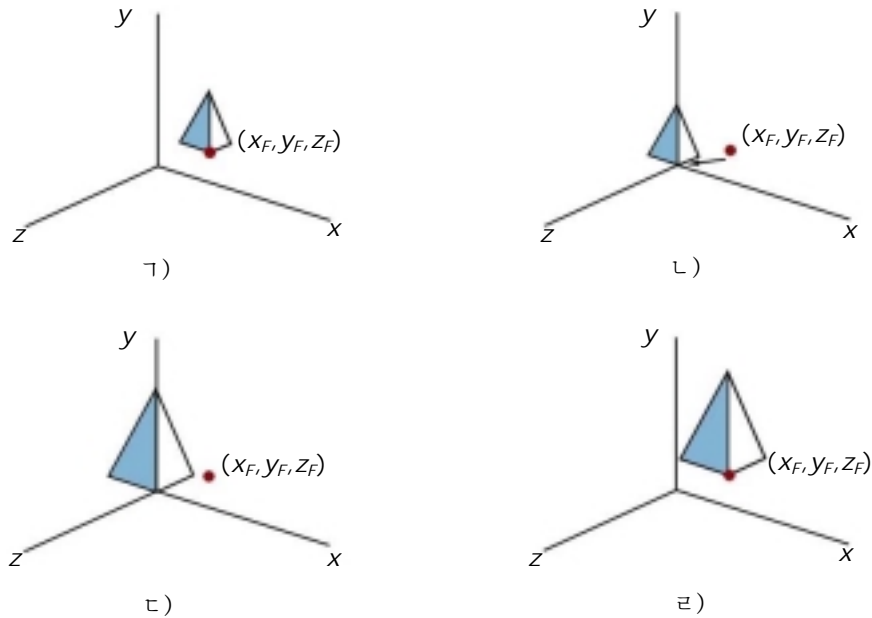



그림 11-18. 선택된 고정점에 대한 물체의 비례변환을 보여 주는 변환렬

식 11-42 또는 11-45에 대한 역비례변환행렬은 비례결수 s_x, s_y, s_z를 그의 역수로 교체하여 얻는다. 역행렬은 반대비례변환을 만들며 임의의 비례변환행렬과 그의 역비례변환행렬의 적은 단위행렬을 만든다.

4절. 기타 변환

평행이동, 회전, 비례변환외에 3차원도형처리에 쓰이는 여러가지 추가적인 변환들이 있다. 그중 두개는 반사와 쏠림이다.

반사

3차원반사는 선택된 반사축 또는 선택된 반사면에 대하여 수행된다. 일반적으로 3차원반사행렬은 2차원과 류사한 방법으로 설정된다. 주어 진 축에 대한 반사는 그 축에 대한 180° 회전과 같다. 평면에 대한 반사는 4차원공간에서의 180° 회전과 같다. 반사면이 자리표평면(xy, xz 또는 yz)일 때 반사변환은 왼손자리표계와 오른손자리표계사이의 변환으로 생각할수 있다.

자리표지적을 오른손자리표계로부터 왼손자리표계에로(또는 거꾸로) 변환하는 반사의 실례를 그림 11-19에 보여 주었다. 이 변환은 z자리표의 부호는 변화시키고 x 및 y자리표의 부호는 변화시키지 않는다. xy평면에 대한 반사행렬은

$$RF_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11-46)$$

이다.

류사하게 x 및 y자리표의 부호를 변화시키는 변환행렬들은 각각 yz평면 및 xz평면에 대한 반사로 정의된다. 기타 평면에 대한 반사는 회전 및 자리표평면반사의 결합으로 얻을수 있다.

쏠림

쏠림변환은 물체의 형태를 수정하는데 리용할수 있으며 3차원보기에서 일반투영변환을 얻는데도 쓸수 있다. 2차원에서 물체의 형태를 변형시키는 x 또는 y축에 대한 쏠림변환을 설명하였다. 3차원에서는 또한 z축에 대한 쏠림을 얻을수 있다.

3차원쏠림의 실례로서 다음의 변환은 z축쏠림을 만든다.

$$SH_z = \begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11-47)$$

파라메터 a와 b에는 임의의 실수값을 줄수 있다. 이 변환행렬은 x 및 y자리표값을 z자리표값에 비례하는 량만큼 변화시키고 한편 z자리표값은 변화시키지 않는다. 그러므로 z축에 수직인 부분평면은 z에 비례하는 량만큼 밀려 진다. 쏠림값 a=b=1인 쏠림행렬을 단위바른6면체에 적용한 실례를 그림 11-20 에 보여 주었다. x 및 y축에 대한 쏠림행렬들도 류사하게 정의된다.

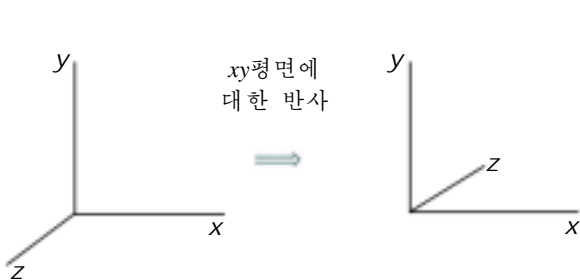



그림 11-19. 반사변환식 11-46 에 의하여 오른손자리표계로부터 왼손자리표계에로 자리표지적을 변환할수 있다.

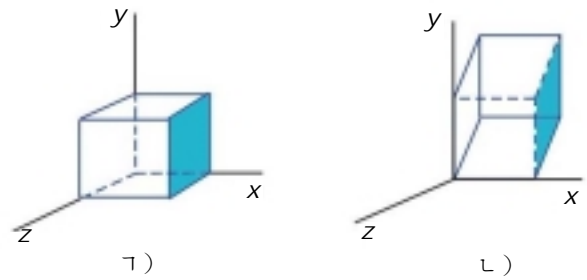



그림 11-20. 단위바른6면체(ㄱ)는 변환행렬 11-47 ($a = b = 1$)에 의하여 쏠려 진다(ㄴ).

5절. 합성변환

2차원변환에서와 마찬가지로 변환렬의 개별적인 변환들에 대한 행렬표현을 곱하여 3차원합성변환을 만든다. 이 련결은 오른쪽에서 왼쪽으로 가면서 수행되며 여기서 제일 오른쪽 행렬은 물체에 적용될 첫번째 변환행렬이며 제일 왼쪽 행렬은 제일 마지막변환행렬이다. 다음의 프로그람은 합성변환의 수행실례를 주었다. 기본적인 3차원기하학적변환들을 결합하여 하나의 합성변환을 만들고 그것을 물체의 자리표정의에 적용한다.

```
#include <math.h>
#include "graphics.h"

#define PI 3.14159

typedef float Matrix4x4[4][4];

Matrix4x4 theMatrix;

void matrix4x4SetIdentity (Matrix4x4 m)
{
  int r,c;

  for (r=0; r<4; r++)
    for (c=0; c<4; c++)
      m[r][c] = (r == c);
}

/* Multiplies matrix a times b, putting result in b */
void matrix4x4PreMultiply (Matrix4x4 a, Matrix4x4 b)
{
  int r,c;
  Matrix4x4 tmp;

  for (r=0; r<4; r++)
    for (c=0; c<4; c++)
      tmp[r][c] = a[r][0]*b[0][c] + a[r][1]*b[1][c] +
        a[r][2]*b[2][c] + a[r][3]*b[3][c];
  for (r=0; r<4; r++)
    for (c=0; c<4; c++)
      b[r][c] = tmp[r][c];
}

void translate3 (float tx, float ty, float tz)
{
  Matrix4x4 m;

  matrix4x4SetIdentity (m);
  m[0][3] = tx; m[1][3] = ty; m[2][3] = tz;
  matrix4x4PreMultiply (m, theMatrix);
}

void scale3 (float sx, float sy, float sz, wcPt3 center)
{
  Matrix4x4 m;

  matrix4x4SetIdentity (m);
```

```
  m[0][0] = sx;
  m[0][3] = (1 - sx) * center.x;
  m[1][1] = sy;
  m[1][3] = (1 - sy) * center.y;
  m[2][2] = sz;
  m[2][3] = (1 - sz) * center.z;
  matrix4x4PreMultiply (m, theMatrix);
}

void rotate3 (wcPt3 p1, wcPt3 p2, float radianAngle)
{
  float length = sqrt ((p2.x - p1.x) * (p2.x - p1.x) +
                       (p2.y - p1.y) * (p2.y - p1.y) +
                       (p2.z - p1.z) * (p2.z - p1.z));
  float cosA2 = cosf (radianAngle / 2.0);
  float sinA2 = sinf (radianAngle / 2.0);
  float a = sinA2 * (p2.x - p1.x) / length;
  float b = sinA2 * (p2.y - p1.y) / length;
  float c = sinA2 * (p2.z - p1.z) / length;
  Matrix4x4 m;

  translate3 (-p1.x, -p1.y, -p1.z);
  matrix4x4SetIdentity (m);
  m[0][0] = 1.0 - 2*b*b - 2*c*c;
  m[0][1] = 2*a*b - 2*cosA2*c;
  m[0][2] = 2*a*c + 2*cosA2*b;
  m[1][0] = 2*a*b + 2*cosA2*c;
  m[1][1] = 1.0 - 2*a*a - 2*c*c;
  m[1][2] = 2*b*c - 2*cosA2*a;
  m[2][0] = 2*a*c - 2*cosA2*b;
  m[2][1] = 2*b*c + 2*cosA2*a;
  m[2][2] = 1.0 - 2*a*a - 2*b*b;
  matrix4x4PreMultiply (m, theMatrix);
  translate3 (p1.x, p1.y, p1.z);
}

void transformPoints3 (int nPts, wcPt3 * pts)
{
  int k,j;
  float tmp[3];

  for (k=0; k<nPts; k++) {
    for (j=0; j<3; j++)
      tmp[j] = theMatrix[j][0] * pts[k].x + theMatrix[j][1] * pts[k].y +
        theMatrix[j][2] * pts[k].z + theMatrix[j][3];
    setWcPt3 (&pts[k], tmp[0], tmp[1], tmp[2]);
  }
}
```

```
void main (int argc, char ** argv)
{
  wcPt3 pts[5] = {10,10,0, 100,10,0, 125,50,0, 35,50,0, 10,10,0};
  wcPt3 p1 = { 10,10,0 }, p2 = { 10,10,10 };
  wcPt3 refPt = { 68.0,30.0,0.0 };

  long windowID = openGraphics (*argv, 200, 200);
  setBackground (WHITE);
  setColor (BLUE);
  pPolyline3 (5, pts);

  matrix4x4SetIdentity (theMatrix);
  rotate3 (p1, p2, PI/4.0);
  scale3 (0.75, 0.75, 1.0, refPt);
  translate3 (25, 40, 0);
  transformPoints3 (5, pts);
  setColor (RED);
  pPolyline3 (5, pts);

  sleep (10);
  closeGraphics (windowID);
}
```

6절. 3차원변환함수

모형화변환행렬과 기타 변환행렬들은 5장의 2차원변환에서 취급한것과 류사한 함수들에 의해 설정한다. 중요한 차이는 임의의 자리표축에 대한 회전을 지적할수 있는것이다. 이 함수들은

```
translate3 (translateVector, matrixTranslate)
rotateX (thetaX, xMatrixRotate)
rotateY (thetaY, yMatrixRotate)
rotateZ (thetaZ, zMatrixRotate)
scale3 (scaleVector, matrixScale)
```

이다. 이 함수들은 동차렬벡토르로 표현되는 자리표위치들을 변환하는데 리용할수 있는 4×4변환행렬을 만든다. 파라메터 translateVector는 평행이동거리 t_x, t_y, t_z들의 목록에 대한 지시자이다. 류사하게 파라메터 scaleVector는 3개의 비례결수 s_x, s_y, s_z를 지적한다. 회전행렬과 비례변환행렬들은 물체를 자리표원점에 대하여 변환시킨다.

합성변환은 함수

```
composeMatrix3
buildTransformationMatrix3
composeTransformationMatrix3
```

에 의해 만들수 있다. 이 함수들은 3개의 회전각을 지적할수 있다는것을 제외하면 합성행렬을 설정하

는 2차원함수들과 류사한 파라메터들을 가진다. buildTransformationMatrix3함수와 composeTransformationMatrix3함수에 대한 변환렬의 순서는 2차원에서와 같다. 즉 첫째로 비례변환, 둘째로 회전, 셋째로 평행이동이다.

변환행렬이 주어 지면 지적된 점에 행렬을 함수

```
transformPoint3 (inPoint, matrix, outPoint)
```

에 의하여 적용할수 있다. 구조물을 리용하는 계층적인 구성에 대한 변환은 함수

```
setLocalTransformation3 (matrix, type)
```

에 의하여 설정할수 있다. 여기서 파라메터 matrix는 4×4변환행렬의 원소들을 지적하며 파라메터 type에는 다음의 3개값 *preconcatenate*, *postconcatenate*, *replace*중 하나를 할당할수 있다.

7절. 모형화변환과 자리표변환

지금까지 물체를 주어 진 자리표계안의 한 위치에서 다른 위치로 움직이는 3차원변환을 고찰하였다. 그러나 한 자리표계로부터 다른 자리표계에로의 자리표변환도 많이 리용한다. 실례로 일반3차원보기절차에서는 처음에 세계자리표를 보기자리표로 변환하고 다음에 보기자리표를 장치자리표로 변환한다. 물체를 모형화할 때 물체들은 흔히 국부(모형화)자리표계에서 표현되며 다음에 세계자리표의 장면안에서 물체들의 위치를 다시 정한다. 실례로 개별적인 국부(모형화)자리표계에서 정의되는 책상, 의자, 기타 가구들은 매 가구자리표를 방자리표로 변환시켜 방에 놓을수 있다. 다음에 방은 세계자리표계에서 만들어 지는 더 큰 장면에로 변환될수 있다.

3차원물체에서 다중자리표계와 계층적인 모형화의 리용실례를 그림 11-21에 주었다. 이 그림은 뜨락또르의 운동모의를 보여 준다. 뜨락또르가 움직일 때 뜨락또르자리표계와 앞바퀴자리표계는 세계자리표계에서 움직인다. 앞바퀴는 앞바퀴자리표계에서 회전하며 앞바퀴자리표계는 뜨락또르가 돌 때 뜨락또르자리표계에서 회전한다.

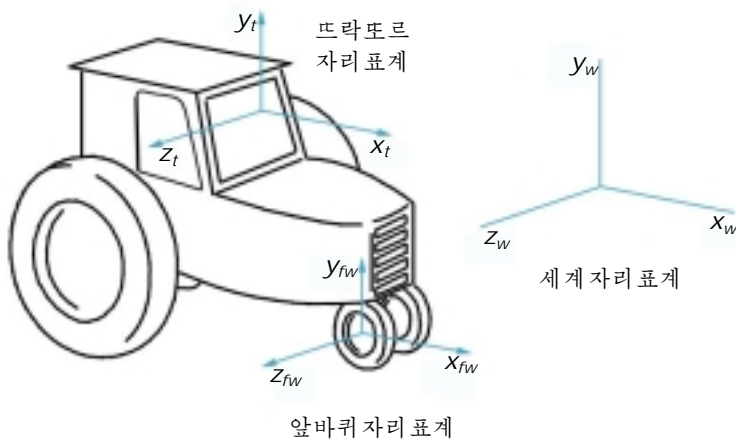



그림 11-21. 뜨락또르운동모의에 리용되는 자리표계들(앞바퀴의 회전은 앞바퀴자리표계에서 표현된다. 앞바퀴자리표계와 뜨락또르자리표계는 다같이 세계자리표계에서 움직인다.)

3차원물체와 장면들은 7장에서 설명한것과 류사한 구조물(또는 토막)연산을 리용하여 만든다. 모형화변환함수들은 3차원물체의 계층적인 표현을 만드는데 적용할수 있다. 3차원물체의 형태는 국부(모형화)자리표계에서 정의할수 있으며 다음에 개별적인 물체들의 구체적인 실례를 가지고 장면이나 물체의 계층적인 표현을 만들수 있다. 즉 물체표현을 모형화자리표계로부터 세계자리표계에로 또는 계층안의 다른 자리표계에로 변환한다. PHIGS구조물계층의 실례를 그림 11-22에 보여 주었다. 이 현시화면은 PHIGS응용을 위한 편집기, 창문, 차림표, 기타 대면도구들을 주기 위하여 만체스터종합대학에서 개발한 PHIGS Toolkit쏘프트웨어로 만들었다. 그림 11-23은 3차원모형화의 두가지 응용실례를 보여 주었다.

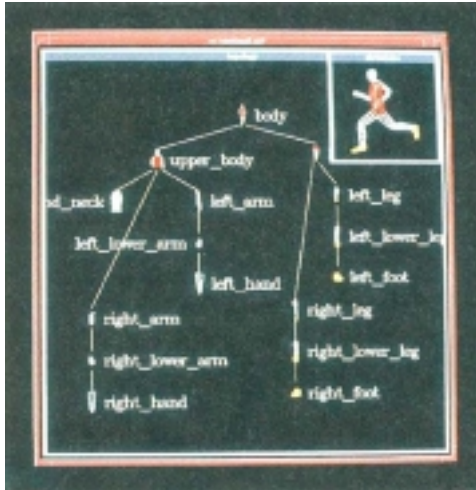



그림 11-22. 만체스터종합대학에서 개발한 PHIGS Toolkit 쏘프트웨어를 리용하는 물체계층의 현시(현시된 물체나무는 그자체가 PHIGS 구조물이다.)

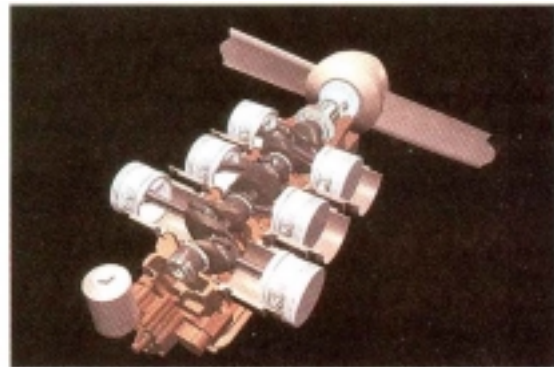
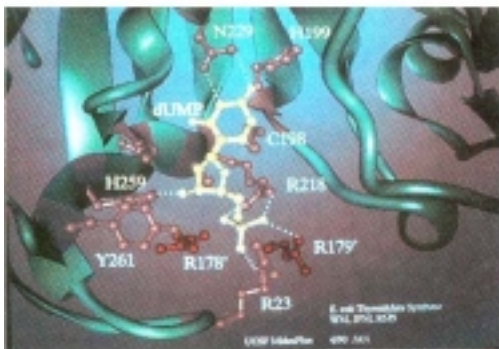



그림 11-23. 3차원모형화

ㄱ-Thymidylate Synthase자연기질과 호상작용하는 key아미노산잔기의 구막대표현, ㄴ-개별적인 기관요소들을 보여 주는 CAD모형

물체의 자리표표현은 2차원자리표변환과 같은 절차로 한 자리표계로부터 다른 자리표계에로 변환된다. 두 자리표계를 일치시키는 변환행렬을 만들어야 한다. 먼저 새 자리표원점을 낡은 자리표원점위치에로 가져 가는 평행이동을 진행한다.다음 자리표축들을 대응시키는 회전들을 진행한다. 두 자리표계의 척도가 다르면 거리에서의 차이를 보상하기 위하여 비례변환을 진행한다.

그림 11-24에서 보여 주는바와 같이 첫번째 자리표계에 대하여 두번째 자리표계가 자리표원점 (x_0, y_0, z_0)과 단위축벡토르들에 의하여 정의되면 먼저 평행이동행렬 $\mathbf{T}(-x_0, -y_0, -z_0)$을 만든다. 다음에 자리표회전행렬

$$\mathbf{R} = \begin{bmatrix} u'_{x1} & u'_{x2} & u'_{x3} & 0 \\ u'_{y1} & u'_{y2} & u'_{y3} & 0 \\ u'_{z1} & u'_{z2} & u'_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11-48)$$

을 만들기 위하여 단위축벡토르들을 리용한다. 이 행렬은 단위축벡토르 $\mathbf{u}'_x$, $\mathbf{u}'_y$, $\mathbf{u}'_z$를 각각 x, y, z축에로 변환한다. 완전한 자리표변환렬은 합성행렬 $\mathbf{R} \cdot \mathbf{T}$에 의하여 주어 진다. 이 행렬은 한 자리표계가 왼손자리표계이고 다른 자리표계가 오른손자리표계라고 하여도 자리표표현을 한 자리표계로부터 다른 자리표계에로 정확히 변환한다.

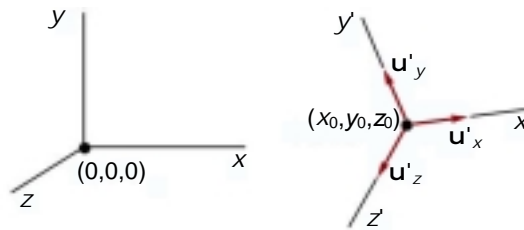



그림 11-24. 물체표현을 한 자리표계로부터 다른 자리표계에로 변환

요약

콤퓨터도형처리에서 널리 쓰이는 3차원변환에는 한 자리표계안에서의 기하학적변환과 자리표계들사이의 변환이 있다. 기본적인 기하학적변환은 평행이동, 회전, 비례변환이다. 그외에 반사와 쏠림의 두가지 변환이 있다. 자리표계들사이의 변환은 모형화루틴과 보기루틴들에서 공통적으로 리용된다. 3차원변환은 4×4행렬로 표현한다. 2차원도형처리에서와 마찬가지로 3차원합성변환은 전체 변환의 개별적인 성분들에 대한 행렬표현을 련결하여 얻는다.

3차원의 평행이동과 비례변환은 2차원변환들의 간단한 확장이다. 그러나 회전인 경우 물체는 공간의 임의의 축에 대하여 회전할수 있기때문에 더 일반적인 표현이 요구된다. 임의의 3차원회전은 기본적인 x, y, z축회전들의 결합으로 표현할수 있다. 대부분의 도형처리프로그람들은 3개의 회전함수들을 준다. 그러나 일반적으로 국부회전축자리표계나 4원수표현을 리용하여 3차원회전을 수행하는것이 더 능률적이다. 특히 4원수는 동화에서 자주 요구되는 반복회전을 빨리 수행하는데 편리하다.

3차원의 반사와 쏠림은 공간의 임의의 자리표축에 대하여 수행할수 있다. 그러므로 이 변환들은 2차원의 대응하는 변환들보다도 더 많이 쓰인다. 물체표현을 한 자리표계로부터 다른 자리표계에로 변환하는것은 두 자리표계가 일치하도록 하는 변환과 등가이다. 마지막으로 물체모형화는 물체의 개별적인 요소들이 전체적인 구조물과 조화롭게 움직이도록 하는 계층적인 변환구조를 요구한다.

참고문헌

행렬, 모형화, 3차원변환의 추가적인 방법들은 Glassner(1990), Arvo(1991), Kirk(1992)에서 보시오. 4원수회전은 Shoemake(1985)에서 구체적으로 고찰하고 있다. 3차원의 PHIGS 및 PHIGS+변환함수들은 Howard 등(1991), Gaskins(1992), Blake(1993)에서 주고 있다.

련습문제

11-1. 다음의 변환렬에 대하여 3차원변환행렬들의 곱하기순서를 바꿀수 있다는것을 증명하시오.

ㄱ. 임의의 련속된 두개의 평행이동

ㄴ. 임의의 련속된 두개의 비례변환

ㄷ. 임의의 자리표축에 대한 임의의 련속된 두개의 회전

11-2. 식 11-30 또는 식 11-41을 리용하여 주어 진 회전축에 대한 임의의 련속된 두 회전은 순서를 바꿀수 있다는것을 증명하시오.

11-3. 식 11-39의 항들을 계산하여 식 11-40에 주어 진 일반회전행렬의 원소들을 유도하시오.

11-4. 식 11-35의 회전행렬은 합성행렬 $\mathbf{R}_y(\beta)\cdot\mathbf{R}_x(\alpha)$와 같다는것을 보여 주시오.

11-5. 회전축이 z축일 때 식 11-40의 4원수회전행렬은 식 11-5의 행렬표현과 같아 진다는것을 증명하시오.

11-6. 식 11-41은 식 11-30에 주어 진 일반회전변환과 같다는것을 증명하시오.

11-7. 식 11-35의 회전행렬을 리용하여 일반회전변환을 수행하는 절차를 쓰시오.

11-8. 임의로 지적된 축에 대하여 식 11-41의 4원수회전을 수행하는 루틴을 쓰시오.

11-9. 방향각 α, β, γ에 의하여 정의되는 방향에서 비례결수 s에 의하여 물체를 비례변환하는 변환행렬을 유도하시오.

11-10. 8분나무표현으로 정의되는 물체를 비례변환하는 알고리듬을 개발하시오.

11-11. 임의로 지적된 축에 대하여 물체를 증분적으로 회전시켜 움직이게 하는 절차를 개발하시오. 계산속도를 높이기 위하여 삼각함수계산에 적당한 근사식을 리용하시오. 주어 진 축에 대한 옹근 한 회전후에는 물체를 자기의 초기위치에 재설정하시오.

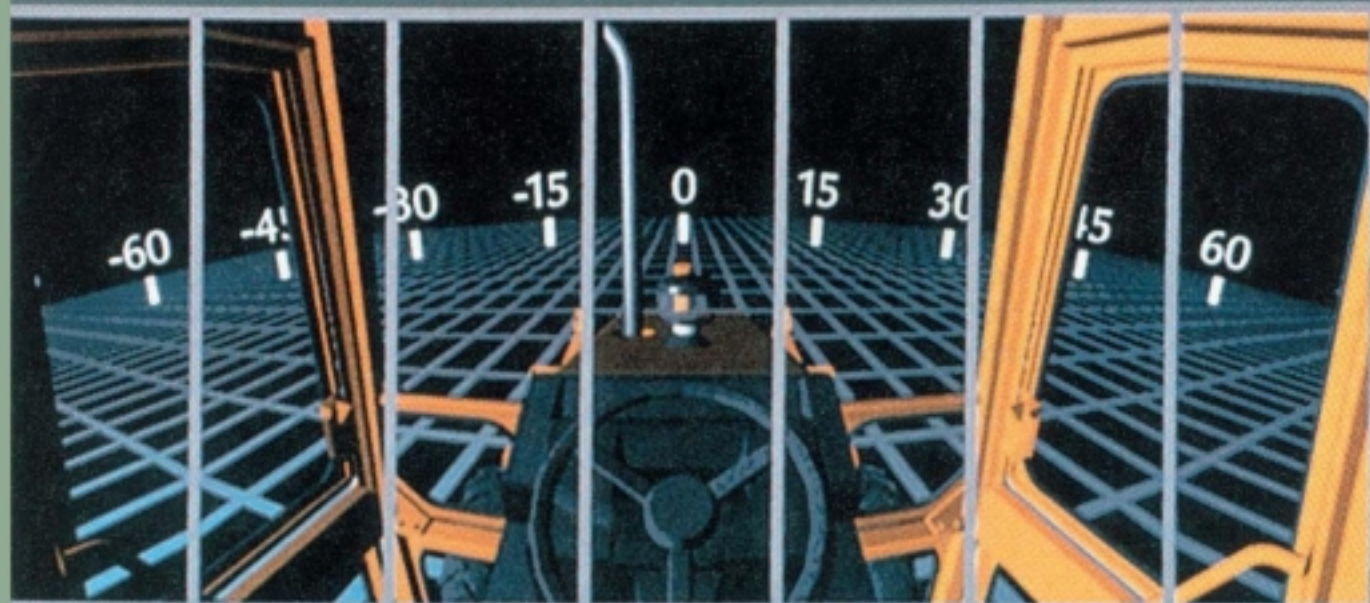
11-12. 8분나무구조로 표현되는 물체를 회전시키는 절차를 만드시오.

11-13. 임의로 선택된 평면에 대하여 물체를 반사시키는 루틴을 개발하시오.

11-14. 쏠림파라메터들의 입력값을 가지고 임의의 자리표축에 대하여 물체를 쏠리게 하는 프로그람을 작성하시오.

11-15. 한 자리표계에서의 물체정의를 그 자리표계에 대하여 정의되는 다른 자리표계에로 변환하는 절차를 개발하시오.

11-16. 3차원기초요소들을 결합하여 새로운 형태를 만드는 립체구성모형화절차의 완전한 알고리듬을 개발하시오. 처음에 기초요소들을 결합하여 부분조립품을 만들고 다음에 부분조립품들을 서로 결합하고 또 기초요소형태들과도 결합하여 최종조립품을 만들수 있다. 평행이동파라메터와 회전파라메터들을 대화식으로 입력하여 물체의 위치를 정할수 있다. 알고리듬의 출력은 최종물체를 만드는데 필요한 조작렬이다.

12장. 3차원보기

2차원도형처리에서 보기조작은 세계자리표면에서의 위치를 출력장치자리표면에서의 화소위치로 변환한다. 세계자리표창문의 직4각형경계와 장치보임창의 직4각형경계를 리용하여 2차원프로그람은 장면의 세계자리표표현을 장치자리표표현으로 넘기며 장면을 보임창의 4경계에 대하여 자른다. 3차원도형처리인 경우 각이한 방향에서 여러가지 방법으로 물체를 볼수 있기때문에 좀 더 복잡하다. 우선 물체를 임의의 공간위치 즉 앞에서, 우에서 또는 뒤에서 볼수 있다. 또한 물체들의 중간에 또는 건물과 같은 단일한 물체의 내부에 서서 볼수 있다. 추가적으로 물체의 3차원표현은 출력장치의 보기면에 투영되여야 한다. 그리고 자르기경계는 공간의 체적을 둘러 싸며 체적의 형태는 선택하는 투영의 류형에 관계된다. 이 장에서는 3차원장면을 보는데 필요한 일반적인 조작들을 연구하며 또한 PHIGS 및 GL과 같은 프로그람들이 제공하는 구체적인 보기절차들을 설명한다.

1절. 보기과정의 흐름

콤퓨터에 의하여 3차원장면을 보는 걸음들은 사진을 찍을 때의 과정들과 약간 류사하다. 순간촬영사진을 찍자면 공간의 특정한 점에 카메라의 위치를 정하여야 한다. 다음에 카메라의 방향을 결정하여야 한다(그림12-1). 즉 카메라가 어느쪽을 가리키도록 하며 그림의 방향을 설정하기 위하여 카메라를 조준선에 대하여 얼마나 회전시켜야 하는가를 결정하여야 한다. 마지막에 샤타를 누를 때 장면은 카메라의 《창문》(시야조절장치)크기로 잘라 지며 보이는 면들에서 나오는 빛은 카메라필림에 투영된다. 그러나 카메라와 류사한것은 여기까지만이다. 왜냐하면 도형처리프로그람에 의하여 장면의 보임상을 만드는것은 카메라로 하는것보다 더 많은 적응성과 수많은 선택항목들을 가지기때문이다.

그림 12-2는 장면의 세계자리표표현을 모형화자리표표현과 장치자리표표현으로 변환하는 일반적인 처리걸음들을 보여 준다. 장면을 모형화한후 세계자리표는 보기자리표로 변환된다. 도형처리프로그람들에서는 관찰자의 보기위치와 함께 카메라필림면과 류사하게 생각할수 있는 투영면의 위치를 지적하기 위한 자리표계로서 보기자리표계를 리용한다. 다음에 투영조작은 장면의 보기자리표표현을 투영면에서의 자리표표현으로 변환하며 그후 투영자리표는 출력장치자리표로 변환된다. 지적된 보기범위밖의 물체들은 앞으로 고찰하지 않기 위하여 자르며 나머지 물체들은 장치보임창안에 현시하기 위하여 보이는 면의 식별과 면렌더링절차를 통해 처리한다.

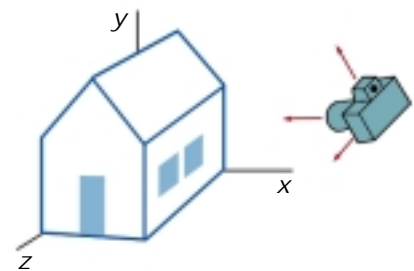



그림 12-1. 장면을 사진찍기 위하여 카메라의 위치와 방향을 선택한다.

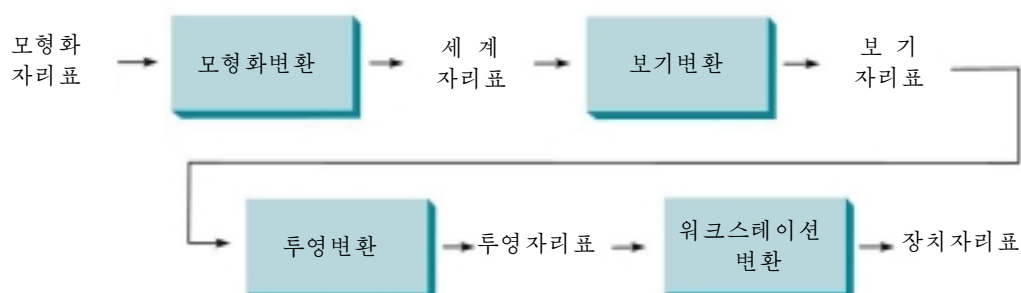



그림 12-2. 모형화자리표로부터 마지막장치자리표에로의 일반적인 3차원변환경로

2절. 보기자리표

3차원에서 물체의 보임상을 만드는것은 물체를 사진 찍는것과 류사하다. 우리는 물체의 주위를 따라 걸으면서 임의의 각도로 각이한 거리에서 카메라의 방향을 여러가지로 변화시켜 물체를 사진 찍을수 있다. 카메라의 시창에 나타나는 모든것은 필림면에 투영된다. 카메라렌즈의 류형과 크기는 마지막사진에 나타나는 부분장면을 결정한다. 이것은 《카메라》의 공간적인 위치, 방향, 시야조절장치의 크기가 주어 지면 장면의 보임상을 만들수 있도록 3차원도형처리프로그람들에서 구체화된다.

보기면의 지적

그림 12-3에 보여 준바와 같이 먼저 보기자리표계를 설정하여 장면의 특정한 보임상을 선택한다. 보기면(view plane) 또는 투영면(projection plane)은 보기자리표계의 z_v축에 수직으로 설정된다. 보기면은 장면의 특정한 촬영을 위하여 위치와 방향이 정해 진 카메라의 필림면으로 생각할수 있다. 장면안의 세계자리표는 보기자리표로 변환되며 다음에 보기자리표는 보기면에 투영된다.

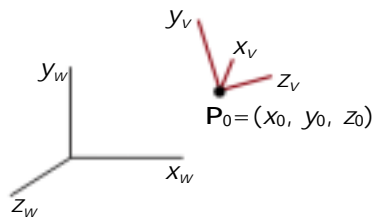



그림 12-3. 세계자리표장면에 대하여 축 x_v, y_v, z_v를 가지는 오른손보기자리표계

보기자리표계를 설정하기 위하여 먼저 보기참조점(view reference point)이라고 하는 세계자리표위치를 고른다. 이 점은 보기자리표계의 원점이다. 보기참조점은 대체로 장면의 일부 물체의 가까이에서나 또는 그의 걸면에서 선택된다. 그러나 물체의 중심점이나 물체들의 무리의 중심점 또는 현시될 장면앞의 임의의 점을 선택할수도 있다. 물체에 가깝거나 또는 물체에 있는 점을 선택하면 이 점은 물체를 사진 찍기 위하여 카메라를 겨누려고 하는 위치로 생각할수 있다. 한편 장면으로부터 어떤 거리에 있는 점을 선택하면 이것은 카메라의 위치로 생각할수 있다.

다음 보기면의 법선벡토르 **N**을 지적하여 보기자리표계의 z_v축에 대한 정의 방향과 보기면의 방향을 선택한다. 세계자리표원점 또는 보기자리표원점에 대하여 **N**의 방향을 설정하는 세계자리표위치를 선택한다. 그림 12-4에 보여 주는바와 같이 실례로 GKS 및 PHIGS와 같은 도형처리프로그람들은 세계자리표원점에 대하여 **N**의 방향을 정한다. 그러면 보기면의 법선 **N**은 세계자리표원점으로부터 선택된 자리표위치에로 향하는 방향선분이다. 다시말하여 **N**은 간단히 세계자리표벡토르로 지적된다. 다른 프로그람(실례로 씰리콘그라픽스의 GL)들은 보기참조점(보기자리표원점)에 대한 고찰점(look-at point)으로 선택되는 자리표위치를 리용하여 **N**의 방향을 설정한다. 그림 12-5는 **N**의 방향을 정하는 이 방법을 보여 주는데 **N**은 고찰점으로부터 보기참조점에로 향한다. 반대로 왼손보기자리표계를 설정하고 보기자리표원점으로부터 고찰점에로 향하는 **N**과 정의 z_v축을 취할수도 있다. z_v의 방향을 설정하는데는 **N**의 방향만 요구되고 그의 크기는 중요하지 않다. 왜냐하면 **N**은 보기계산에 의해 단위벡토르로 정규화되기때문이다.

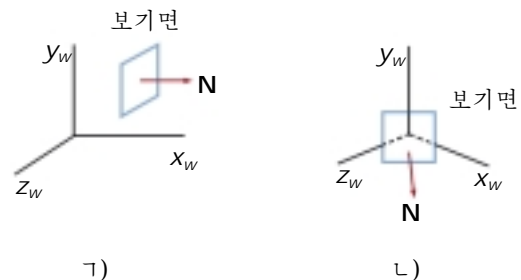



그림 12-4. 세계자리표원점에 대하여 지적된 법선벡토르의 자리표에 의한 보기면의 방향(위치 (1,0,0)은 ㄱ에서와 같이 보기면의 방향을 정하며 한편 (1,0,1)은 ㄴ의 보기면의 방향을 준다.)

마지막으로 보기의 웃방향벡토르(view-up vector)라고 하는 벡토르 **V**를 지적하여 보기의 웃방향을 선택한다. 이 벡토르는 y_v축에 대한 정의 방향을 설정하는데 리용된다. 벡토르 **V**는 세계자리표벡토르로 정의할수도 있고 또는 그림 12-6에 보여주는바와 같이 일부 프로그람들에서 z_v축에 대한 회전각 θ_t로 지적할수도 있다. 임의의 방향의 법선벡토르 **N**에 정확히 수직인 **V**의 방향을 결정하는것은 힘들수 있다(적어도 시간이 걸릴것이다.). 따라서 보기절차들은 일반적으로 그림 12-7에 보여 준바와 같이 **V**가 법선벡토르에 수직인 면에 투영되도록 사용자가 정의한 벡토르 **V**의 방향을 조정한다. 보기의 웃방향벡토르 **V**는 **N**에 평행이 아닌 임의의 편리한 방향으로 선택할 수 있다. 실례로 PHIGS를 리용하여 보기자리표계를 대화식으로 지적하는것을 고찰하자. 여기서 보기참조점은 보려는 물체의 중심에 설정된다. 그림 12-8에 보여 준 모가 난 방향으로 물체를 보려 한다면 간단히 **V**를 세계자리표벡토르 (0, 1, 0)으로 선택할수 있으며 이 벡토르는 y_v축을 설정하기 위하여 **N**에 수직인 면에로 투영될것이다. 이 방법은 **N**에 정확히 수직인 벡토르를 입력하는것보다 훨씬 더 쉽다.

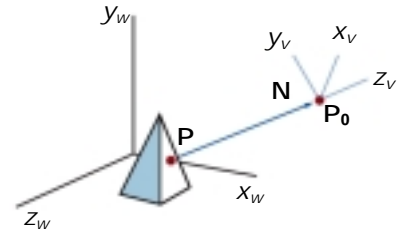



그림 12-5. 보기자리표원점 $\mathbf{P}_0$에 대하여 지적된 고찰점 **P**에 의한 보기면의 방향

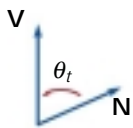



그림 12-6. 회전각 θ_t에 의하여 보기의 웃방향벡토르를 지적

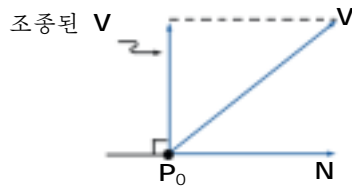



그림 12-7. 입력된 보기의 웃방향벡토르의 방향을 법선벡토르 **N**에 수직인 방향으로 조정

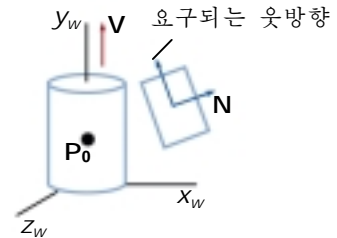



그림 12-8. y_w축을 따라 **V**를 선택하여 보기면의 웃방향을 요구되는 방향으로 설정한다.

도형처리프로그람은 x_v축에 대한 방향을 정의하기 위하여 벡토르 **N**과 **V**를 리용하여 이 두 벡토르에 다같이 수직인 세번째 벡토르 **U**를 계산할수 있다. 다음에 **V**의 방향은 y_v보기자리표축의 방향을 설정하기 위하여 **N**과 **U**에 다같이 수직이 되도록 조정될수 있다. 다음 소제목(세계자리표의 보기자리표에로의 변환)에서 보게 되는바와 같이 단위축벡토르들을 리용하면 이 계산들을 편리하게 할수 있다. 또한 단위축벡토르들은 세계-보기자리표변환행렬의 원소들을 얻는데도 리용할수 있다. 보기자리표계를 자주 *uvn*자리표계로 표현하기도 한다(그림 12-9).

일반적으로 도형처리프로그람들은 보기자리표원점으로부터 보기면까지의 거리(보기면거리)를 지적하여 사용자가 z_v축을 따라 보기면의 위치를 선택(제한적으로)할수 있게 한다. 보기면은 항상 x_vy_v평면에 평행이며 보기면에 대한 물체들의 투영은 출력장치에 현시될 장면의 보임상에 대응한다. 그림 12-10에서는 보기면의 위치를 정하는 한가지 실례를 보여 주었다. 보기면거리를 값 0으로 설

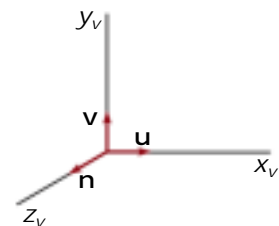



그림 12-9. 단위벡토르 **u, v, n**에 의하여 정의되는 오른손보기자리표계

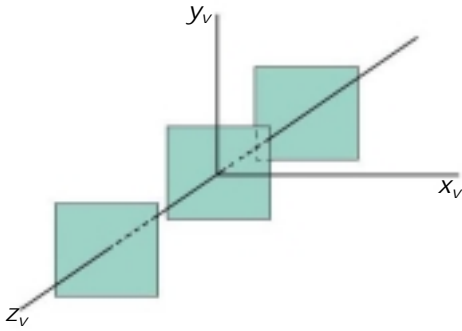



그림 12-10. z_v 축을 따라 보기면의 위치지정

정하면 보기자리표계의 x_vy_v평면(또는 uv평면)은 투영변환을 위한 보기면으로 된다. 때때로 용어 《uv평면》은 보기면이 x_vy_v평면에 대하여 어디에 위치 정해 지는가에 관계없이 보기면이라고도 한다. 그러나《uv평면》이라는 말은 x_vy_v평면을 의미하는데만 리용하려고 하며 그것은 꼭 보기면이 아니다.

도형처리프로그람들에서는 보기방향을 정의 z_v축방향으로 하기 위하여 왼손보기자리표계를 리용한다. 그러나 오른손보기자리표계는 세계자리표계와 꼭같은 자리표축방향을 가지기때문에 보다 일반적이다. 이것은 도형처리체계들이 세계자리표와 보기자리표에 대하여 다같이 하나의 자리표방향만을 다룰수 있게 한다. PHIGS와 GL의 협약에 따라 모든 알고리듬개발에 오른손보기자리표계를 리용하겠다.

장면의 여러가지 보임상을 얻기 위하여 그림 12-11에 보여 준바와 같이 보기참조점을 고정시키고 **N**의 방향을 변화시킬수 있다. 이것은 보기자리표원점주위로 움직일 때의 보임상을 만드는데 대응한다. 대화식응용들에서 법선벡토르 **N**은 제일 많이 변화되는 보기파라메터이다. 보기의 웃방향에 따라 보는것을 제외하면 **N**의 방향만을 변화시켜 임의의 방향에서 장면을 볼수 있다. 보기의 웃방향에 따라 가능한 두개의 보임상을 얻자면 **V**의 방향을 변화시켜야 한다. 장면을 따라 카메라의 운동을 모의하려면 **N**을 고정시키고 보기참조점을 사방으로 움직여야 한다(그림 12-12).

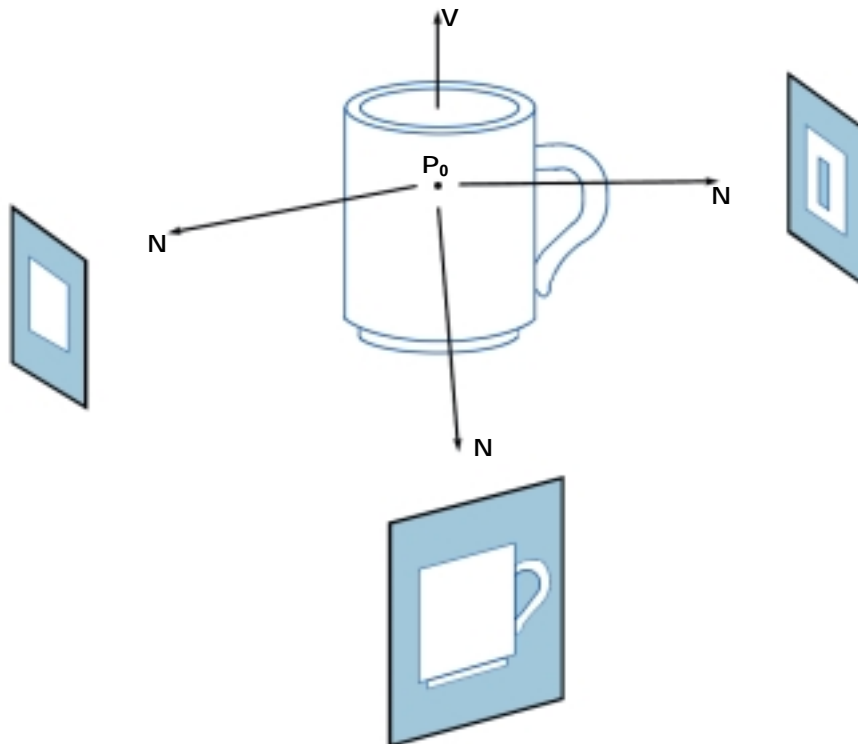



그림 12-11. 보기참조점을 고정시키고 여러 방향에서 장면보기

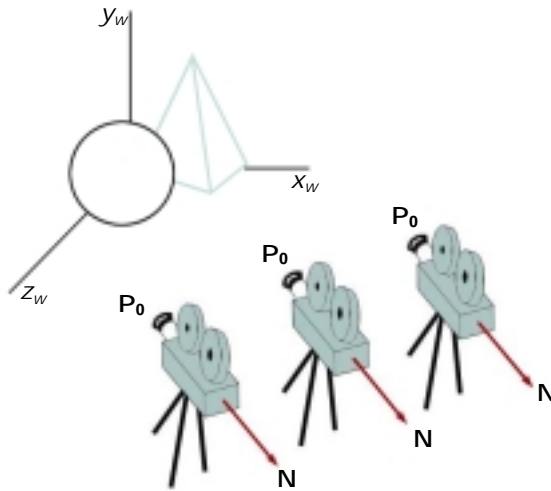



그림 12-12. 보기참조점의 위치를 변화시켜 장면에서 사방으로 이동

세계자리표의 보기자리표에로의 변환

물체의 세계자리표표현은 보기면에로 투영하기전에 보기자리표표현으로 넘겨야 한다. 물체의 세계자리표표현을 보기자리표표현으로 넘기는 변환은 11장 7절에서 설명한 기하학적인 평행이동-회전조작을 리용하여 보기자리표계를 세계자리표계에 덧놓는 변환과 등가이다. 이 변환렬은

1. 보기참조점을 세계자리표원점에로 평행이동시킨다.
2. x_v, y_v, z_v축들을 각각 x_w, y_w, z_w축들과 일치시키기 위하여 회전을 적용한다.

보기참조점이 세계자리표위치 (x_0, y_0, z_0)으로 지적되면 이 점은 행렬변환

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12-1)$$

에 의해 세계자리표원점에로 평행이동된다.

회전렬은 **N**의 선택방향에 따라 자리표축회전을 3번까지 요구할수 있다. 일반적으로 **N**이 어느 세계자리표축과도 평행이 아니면 변환렬 $\mathbf{R}_z \cdot \mathbf{R}_y \cdot \mathbf{R}_x$를 리용하여 보기자리표계를 세계자리표계에 덧놓을수 있다. 즉 먼저 z_v축을 $x_w z_w$평면에 넘기기 위하여 x_w축주위로 회전시킨다. 다음에 z_w축과 z_v축을 일치시키기 위하여 y_w축주위로 회전시킨다. 마지막회전은 y_w축과 y_v축을 일치시키기 위한 z_w축회전이다. 보기자리표계가 왼손자리표계이면 보기자리표축들중 하나(실례로 z_v축)에 대한 반사가 필요하다. 그림 12-13에서는 일반적인 평행이동-회전변환렬을 보여 주었다. 합성변환행렬은 다음에 세계자리표를 보기자리표로 넘기는데 적용된다.

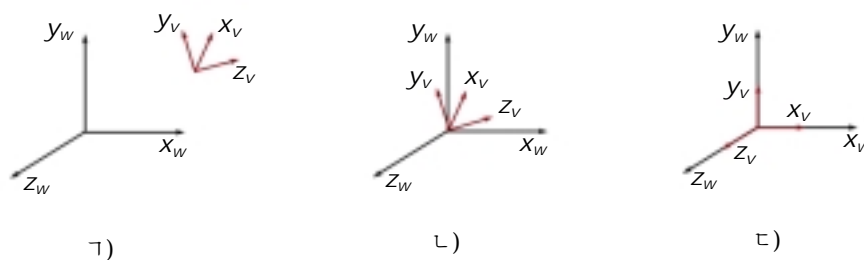



그림 12-13. 평행이동-회전변환렬을 리용하여 보기자리표계를 세계자리표계와 일치시킨다.

회전변환행렬을 만드는 다른 방법은 11장 7절에서 설명한바와 같이 단위 *uvn*벡토르들을 계산하고 합성회전행렬을 직접 만드는것이다. 벡토르 **N**과 **V**가 주어 지면 단위 *uvn*벡토르들은

$$\mathbf{n} = \frac{\mathbf{N}}{|\mathbf{N}|} = (n_1, n_2, n_3)$$
$$\mathbf{u} = \frac{\mathbf{V}\times\mathbf{N}}{|\mathbf{V}\times\mathbf{N}|} = (u_1, u_2, u_3) \qquad (12\text{-}2)$$
$$\mathbf{v} = \mathbf{n}\times\mathbf{u} = (v_1, v_2, v_3)$$

와 같이 계산된다. 이 방법은 **v**가 **n**에 수직이 되도록 **V**의 방향을 자동적으로 조정한다. 그러면 보기변환에 대한 합성회전행렬은

$$\mathbf{R} = \begin{bmatrix} u_1 & u_2 & u_3 & 0 \\ v_1 & v_2 & v_3 & 0 \\ n_1 & n_2 & n_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12-3)$$

이다. 이것은 **u**를 x_w축에로, **v**를 y_w축에로, **n**을 z_w축에로 변환한다. 또한 이 행렬은 왼손보기자리표계를 오른손세계자리표계로 변환하는데 필요한 반사를 자동적으로 수행한다.

완전한 세계-보기자리표변환행렬은 행렬적

$$\mathbf{M}_{WC,VC} = \mathbf{R}\cdot\mathbf{T} \qquad (12\text{-}4)$$

으로 얻어 진다. 이 변환은 다음에 장면안의 물체의 세계자리표표현을 보기자리표표현으로 넘기는데 적용된다.

3절. 투영

장면에서 물체의 세계자리표표현이 보기자리표표현으로 변환되면 3차원물체를 2차원보기면에 투영할수 있다. 두가지 기본투영방법이 있다. 평행투영에서는 그림 12-14의 실례에서 보여 준바와 같이 자리표위치들이 평행선들을 따라 보기면에 넘어 간다. 원근투영(그림 12-15)에서 물체위치들은 투영참조점(또는 투영중심)이라고 하는 점에 집중하는 선들을 따라 보기면에 넘어 간다. 물체의 투영된 보임상은 투영선과 보기면과의 사귐점들을 계산하여 결정한다.

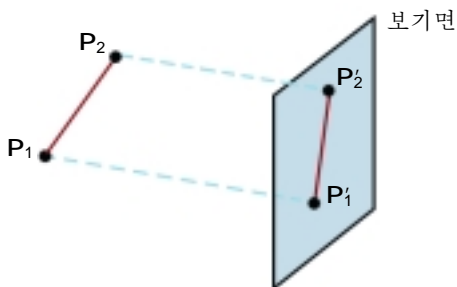



그림 12-14. 보기면에 대한 물체의 평행투영

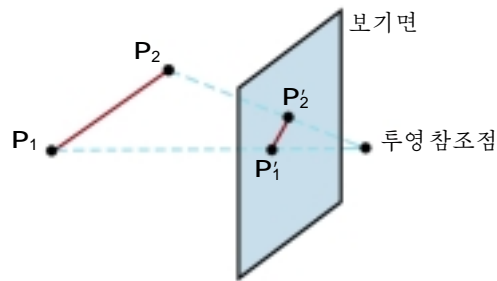



그림 12-15. 보기면에 대한 물체의 원근투영

평행투영은 물체의 상대적인 비례를 보존한다. 제도에서는 3차원물체의 비례변환상을 만들기 위하여 평행투영을 리용한다. 평행투영을 리용하여 물체의 여러가지 측면들을 정밀하게 볼수 있지만 3차원물체를 현실에서 보는것처럼 표현할수는 없다. 반대로 원근투영을 리용하면 물체를 현실에서와 같이 볼수 있지만 물체의 상대적인 비례는 보존할수 없다. 투영면으로부터 먼 거리에 있는 물체의 투영은 가까운 거리에 있는 같은 크기의 물체의 투영보다 더 작다(그림 12-16).

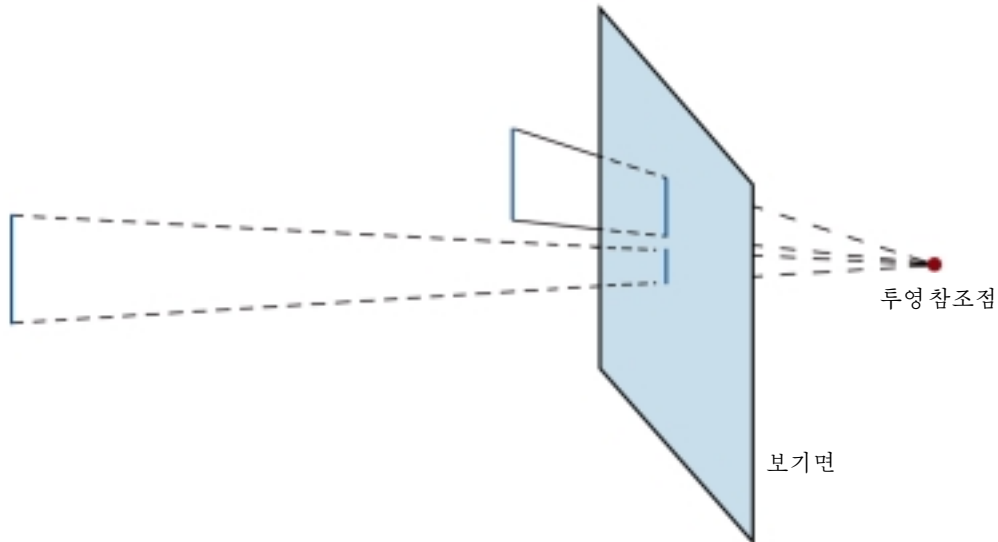



그림 12-16. 보기면으로부터 서로 다른 거리에 있는 같은 크기의 물체들의 원근투영

평행투영

평행투영은 투영선의 방향을 정의하는 투영벡토르에 의하여 지적할수 있다. 투영선이 보기면에 수직일 때 정투영, 그렇지 않을 때 비탈투영이라고 한다. 그림 12-17은 두가지 류형의 평행투영을 보여 주었다. 씰리콘그라픽스 워크스테이션의 GL과 같은 일부 도형처리프로그람들은 비탈투영을 제공하지 않는다. 실례로 이 프로그람에서 평행투영은 간단히 직6면체의 경계들을 주어 지적한다.

그림 12-17. 정투영(ㄱ)과 비탈투영(ㄴ)을 만드는 투영벡토르 $\mathbf{V}_p$의 방향

그림 12-18에 보여 준바와 같이 물체의 앞면도, 옆면도, 웃면도를 만드는데 정투영을 많이 리용한다. 물체를 앞과 옆, 뒤에서 정투영한것은 립면도라고 하며 우에서 정투영한것은 평면도라고 한다. 일반적으로 공학 및 건축그림들은 정투영을 리용한다. 왜냐하면 길이와 각이 정밀하게 그려 지며 그림에서 측정할수 있기때문이다.

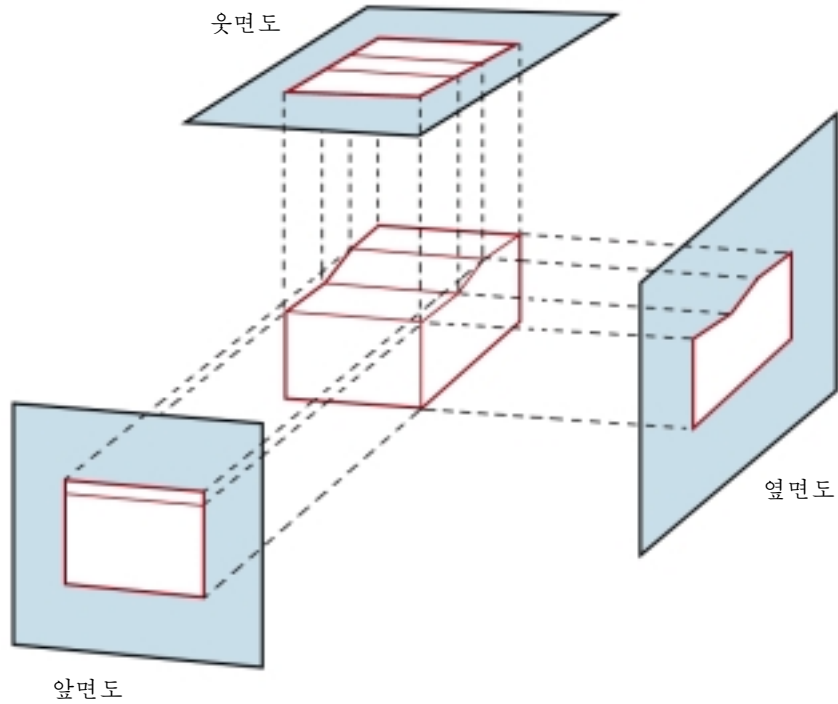



그림 12-18. 물체의 평면도와 립면도를 현시하는 정투영

물체의 하나이상의 면들을 현시하도록 정투영할수 있다. 이러한 투영을 축측투영이라고 한다. 제일 일반적으로 리용하는 축측투영은 등측투영이다. 등측투영은 물체가 정의되는 매개 자리표축(주축이라고 한다.)과 원점으로부터 같은 거리에서 사귀도록 투영면을 설치하여 만든다. 그림 12-19에서는 바른6면체에 대한 등측투영을 보여 주었다. 등측투영은 투영벡토르를 바른6면체의 대각선과 평행되게 하여 얻는다. 등측투영은 매 8분공간에서 하나씩 얻을수 있다. 등측투영에서 3개의 모든 주축들은 상대적인 비례를 유지하기 위하여 꼭같은 원근으로 그려 진다. 이것은 일반적인 축측투영의 경우가 아니다. 일반축측투영에서는 비례결수들이 3개의 주축방향에 대하여 서로 다를수 있다.

정투영의 변환식은 간단하다. 보기면이 z_v축을 따라 위치 z_{vp}에 있으면(그림 12-20) 보기자리표계의 임의의 점 (x, y, z)는

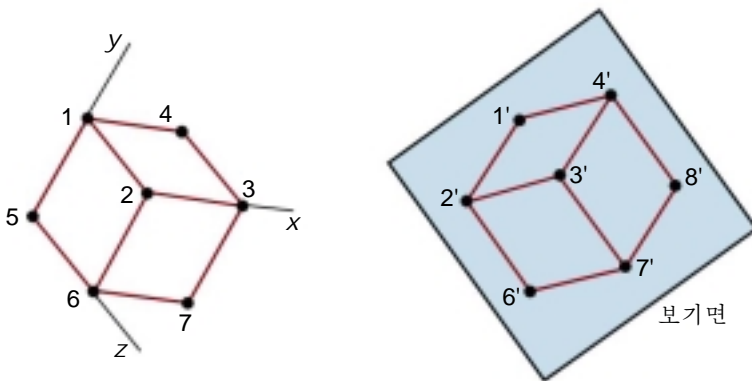



그림 12-19. 바른6면체에 대한 등측투영

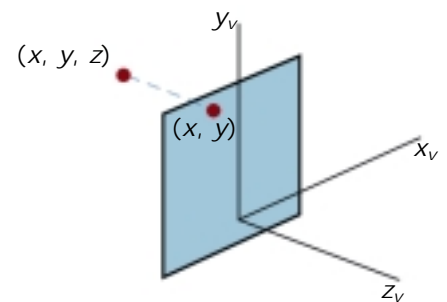



그림 12-20. 보기면에 대한 점의 정투영

$$x_p = x, \qquad y_p = y \tag{12-5}$$

와 같이 투영자리표로 변환된다. 여기서 본래의 z자리표값은 깊이삽입과 보이는 면의 결정절차에 필요한 깊이정보로 보존된다.

비탈투영은 점을 투영면에 수직이 아닌 평행선을 따라 투영하여 얻는다. 일부 응용프로그람들에서 비탈투영벡토르는 그림 12-21에 보여 준바와 같이 두개의 각 α와 β에 의하여 지적된다. 점 (x, y, z)는 보기면의 위치(x_p, y_p)에로 투영된다. 그 평면에서 정투영자리표는 (x, y)이다. (x, y, z)로부터 (x_p, y_p)에로의 비탈투영선은 (x_p, y_p)와 (x, y)를 련결하는 투영면의 선과 각 α를 이룬다. 길이가 L인 이 선은 투영면의 수평방향과 각 ϕ를 이룬다. 투영자리표를 x, y, L, ϕ에 의하여

$$\begin{aligned} x_p &= x + L\cos\phi \\ y_p &= y + L\sin\phi \end{aligned} \tag{12-6}$$

와 같이 표현할수 있다.

길이 L은 각 α와 투영되는 점의 z자리표에 관계된다.

$$\tan\alpha = \frac{z}{L} \tag{12-7}$$

그리하여

$$\begin{aligned} L &= \frac{z}{\tan\alpha} \\ &= zL_1 \end{aligned} \tag{12-8}$$

여기서 L_1은 $\tan\alpha$의 역수이며 또한 z=1일 때의 L의 값이다. 그러면 비탈투영식 12-6을

$$\begin{aligned} x_p &= x + z(L_1\cos\phi) \\ y_p &= y + z(L_1\sin\phi) \end{aligned} \tag{12-9}$$

와 같이 쓸수 있다.

$x_v y_v$평면에로의 임의의 평행투영을 만드는 변환행렬은

$$\mathbf{M}_{\text{parallel}} = \begin{bmatrix} 1 & 0 & L_1\cos\phi & 0 \\ 0 & 1 & L_1\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12-10}$$

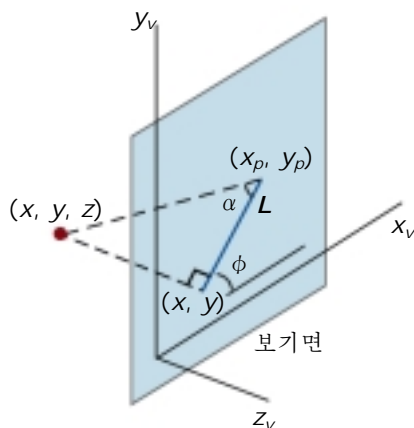



그림 12-21. 자리표위치 (x, y, z)를 보기면우의 위치 (x_p, y_p)에로 비탈투영

와 같이 쓸수 있다. 정투영은 $L_1=0$일 때(투영각 α가 90°일 때) 얻어 진다. 비탈투영은 L_1의 값이 0이 아닐 때 만들어 진다. 투영행렬 12-10은 z축쏠림행렬과 류사한 구조를 가진다. 사실 이 투영행렬의 효과는 z가 상수인 면들을 쏠리게 하고 그것들을 보기면에로 투영하는것이다. z가 상수인 평면안의 x 및 y자리표값들은 그 평면안의 각, 거리, 평행선들이 정밀하게 투영되도록 평면의 z값에 비례하는 량만큼 옮겨 진다. 이 효과를 그림 12-22에 보여 주었다. 여기서 직6면체의 뒤면은 쏠리여 보기면에로의 투영에서 앞면과 겹쳐 진다. 앞면과 뒤면을 련결하는 직6면체의 변은 길이가 L_1인 선으로 투영되며 그것은 투영면의 수평선과 각 ϕ를 이룬다.

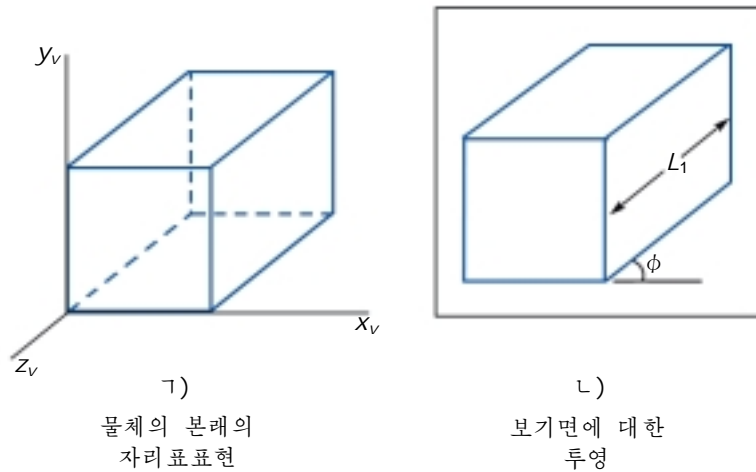



그림 12-22. z_v=0면에 대한 직6면체의 비탈투영

각 ϕ에 대한 일반적인 선택은 30°, 45°이며 그것은 물체의 앞, 옆, 우(또는 앞, 옆, 밑)를 함께 보여 준다. α에 대하여 일반적으로 리용되는 두가지 값은 $\tan\alpha=1$ 및 $\tan\alpha=2$인것들이다. 첫번째 경우 $\alpha=45°$이며 이때의 투영을 캐빌리어(cavalier)투영이라고 한다. 투영면에 수직인 모든 선들은 길이의 변화없이 투영된다. 바른6면체에 대한 캐빌리어투영의 실례를 그림 12-23에 보여 주었다.

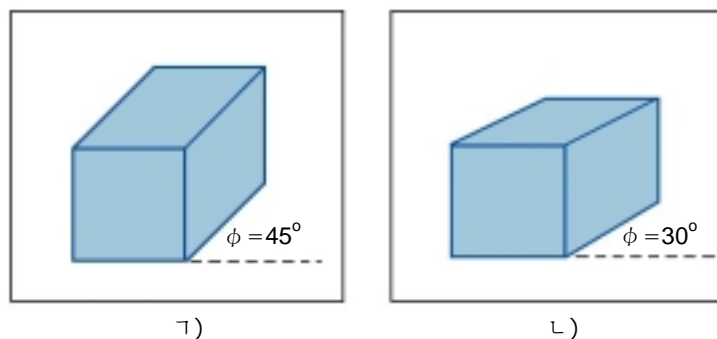



그림 12-23. 각 ϕ의 두 값에 대한 보기면에로의 바른6면체의 캐빌리어 투영(바른6면체의 길이는 너비 및 높이와 꼭같이 투영된다.)

$\tan\alpha=2$가 되도록 투영각 α가 선택될 때의 투영을 캐비니트(cabinet)투영이라고 한다.이 각(≈63.4°)에 대하여 보기면에 수직인 선들은 절반길이로 투영된다. 캐비니트투영은 수직선들에서 길이가

축소되기때문에 캐빌리어투영보다 더 현실감 있게 보인다. 그림 12-24에서는 바른6면체에 대한 캐비니트투영실례를 보여 주었다.

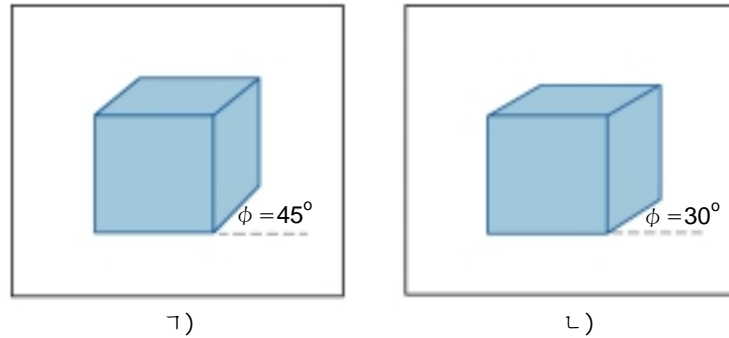



그림 12-24. 각 ϕ의 두 값에 대한 보기면에로의 바른6면체의 캐비니트투영(깊이는 너비와 높이의 절반으로 투영된다.)

원근투영

3차원물체의 원근투영을 얻기 위하여 점들을 투영참조점에서 만나는 투영선들을 따라 변환한다. 그림 12-25에 보여 주는바와 같이 투영참조점을 z_v축을 따라 위치 z_{prp}에 설정하고 보기면을 z_{vp}에 놓는다고 하자. 원근투영선의 방정식을 보조변수형식으로

$$
\begin{aligned}
x' &= x - xu \\
y' &= y - yu \\
z' &= z - (z - z_{prp})u
\end{aligned}
\tag{12-11}
$$

와 같이 쓸수 있다. 파라메터 u는 0∼1사이의 값을 취하며 자리표위치(x', y', z')는 원근투영선우의 임의의 점을 표현한다. $u=0$일 때 위치 $\mathbf{P}=(x, y, z)$에 있다. 원근투영선의 다른 끝에서 $u=1$이며 투영참조점의 자리표$(0, 0, z_{prp})$를 가진다. 보기면에서 $z'=z_{vp}$이며 투영선우의 이 위치에서 파라메터 u에 대하여 z'식을 풀수 있다.

$$
u = \frac{z_{vp} - z}{z_{prp} - z}
\tag{12-12}
$$

u의 이 값을 x'와 y'에 대한 식들에 대입하면 원근변환식

$$
\begin{aligned}
x_p &= x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = x\left(\frac{d_p}{z_{prp} - z}\right) \\
y_p &= y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = y\left(\frac{d_p}{z_{prp} - z}\right)
\end{aligned}
\tag{12-13}
$$

을 얻는다. 여기서 $d_p=z_{prp}-z_{vp}$는 투영참조점으로부터 보기면까지의 거리이다.

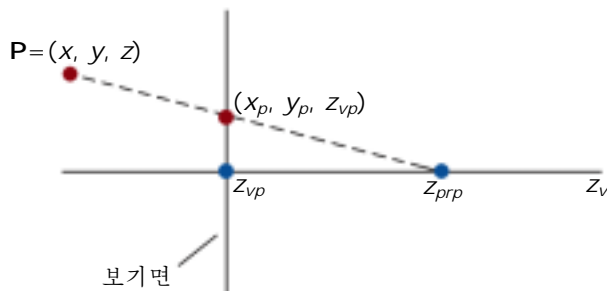



그림 12-25. 자리표가 (x, y, z)인 점을 보기면우의 위치 (x_p, y_p, z_{vp})에로 원근투영

3차원동차자리표표현을 리용하여 원근투영변환식 12-13을 행렬형식으로

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -z_{vp}/d_p & z_{vp}(z_{prp}/d_p) \\ 0 & 0 & -1/d_p & z_{prp}/d_p \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (12-14)$$

와 같이 쓸수 있다. 이 표현에서 동차결수는

$$h = \frac{z_{prp} - z}{d_p} \qquad (12-15)$$

이며 보기면우의 투영자리표는 동차자리표로부터

$$x_p = x_h/h, \qquad y_p = y_h/h \qquad (12-16)$$

와 같이 계산된다. 여기서 본래의 z자리표값은 보이는 면 및 다른 깊이처리를 위하여 투영자리표에 보존될수 있다.

일반적으로 투영참조점은 z_v축을 따라 설정하지 않아도 된다. 보기면의 량쪽에서 임의의 자리표위치 $(x_{prp}, y_{prp}, z_{prp})$에 투영참조점을 설정할수 있으며 이 일반화는 다음절에서 설명한다.

원근변환식 12-13에 대한 몇가지 특수경우가 있다. 보기면이 uv면으로 취해 지면 z_{vp}=0이며 투영자리표는

$$x_p = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = x\left(\frac{1}{1 - z/z_{prp}}\right)$$
$$y_p = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = y\left(\frac{1}{1 - z/z_{prp}}\right) \qquad (12-17)$$

이다. 그리고 일부 도형처리프로그람들에서 투영참조점은 항상 보기자리표원점에 있도록 취해 진다. 이 경우에 z_{prp}=0이며 보기면우의 투영자리표는

$$x_p = x\left(\frac{z_{vp}}{z}\right) = x\left(\frac{1}{z/z_{vp}}\right)$$
$$y_p = y\left(\frac{z_{vp}}{z}\right) = y\left(\frac{1}{z/z_{vp}}\right) \qquad (12-18)$$

이다.

원근변환식을 리용하여 3차원물체를 보기면에 투영할 때 물체에서 보기면에 평행이 아닌 임의의 평행선들은 한점에 모이는 선들로 투영된다. 보기면에 평행인 평행선들은 평행선들로 투영될것이다. 투영된 평행선들이 모이는 점을 소실점이라고 한다. 투영된 평행선들의 매개 모임은 개개의 소실점을 가질것이며 일반적으로 장면은 거기에 평행선들의 모임이 얼마나 많은가에 따라 임의의 개수의 소실점을 가질수 있다.

물체의 주축에 평행인 선들에 대한 소실점을 주소실점이라고 한다. 주소실점의 수는 투영면의 방향에 의하여 조종되며 원근투영은 그에 따라 한점, 두점, 세점투영으로 분류된다. 투영에서 주소실점의 수는 보기면과 사귀는 주축의 수에 의하여 결정된다. 그림 12-26은 바른6면체에서 나타나는 한점 및 두점 원근투영을 설명한다. 그림 12-26 ㄱ에서 보기면은 물체의 z축과만 사귀도록 물체의 xy평면에 평행으로 설치되며 이때 z축소실점을 가지는 한점 원근투영으로 된다. 그림 12-26 ㄴ에서 투영면은 x 및 z축과는 사귀지만 y축과는 사귀지 않는다. 결과 x 및 z축소실점을 가지는 두점 원근투영으로 된다.

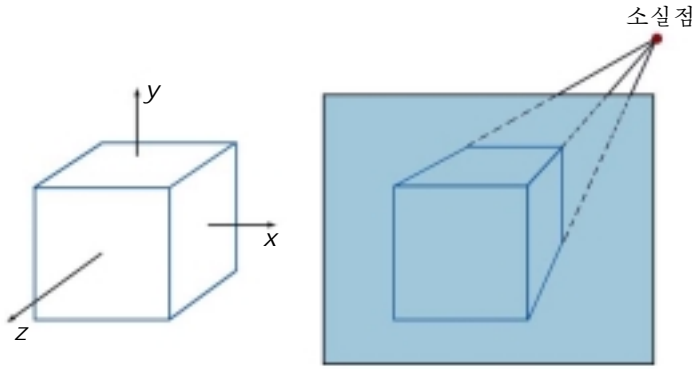
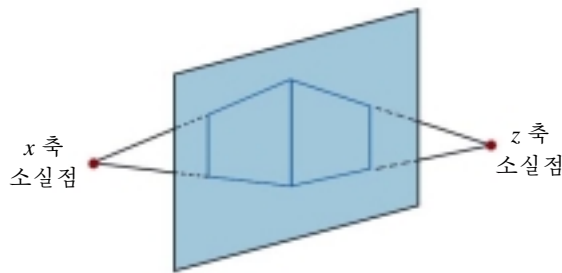



그림 12-26. 물체의 주축들에 대하여 보기면을 여러 방향으로 설치할 때 바른6면체의 원근투영과 주소실점

4절. 보기체적과 일반투영변환

카메라에서 렌즈의 류형은 장면이 필림에 얼마나 많이 잡히는가를 결정하는 하나의 인자이다. 광각렌즈는 정규렌즈보다 장면을 더 많이 잡는다. 3차원보기에서 보기면에 있는 직4각형모양의 보기창문 또는 투영창문은 이와 같은 효과에 리용된다. 보기창문의 변들은 $x_v y_v$축에 평행이며 창문경계의 위치들은 그림 12-27에 보여 준바와 같이 보기자리표로 지적된다. 보기창문은 보기면의 임의의 곳에 놓을수 있다.

보기창문이 지적되면 창문의 경계들을 리용하여 보기체적을 설정할수 있다. 보기체적안의 물체들만이 출력장치에 현시되며 다른것들은 현시되지 않고 모두 잘라진다. 보기체적의 크기는 창문의 크기에 관계되며 한편 보기체적의 형태는 현시에 리용하는 투영의 류형에 관계된다. 보기체적의 네 옆면들은 창문의 변을 지나는 평면이다. 평행투영에서 보기체적의 이 네 옆면들은 그림 12-28에서와 같이 무한평행6면체를 만든다. 원근투영에서 보기체적은 투영참조점에 정점이 있는 각추이다(그림 12-29).

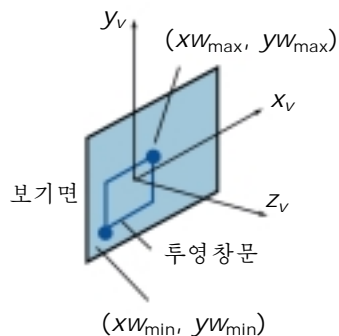



그림 12-27. 보기자리표계에서 최소자리표와 최대자리표를 주어 보기창문을 지적한다.

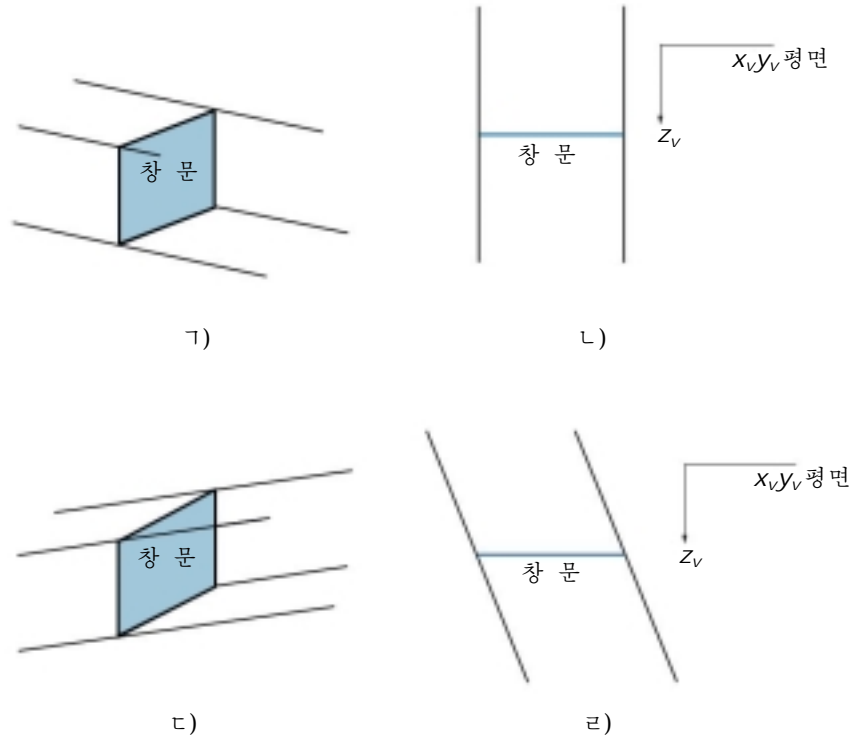



그림 12-28. 평행투영의 보기체적(ㄱ와 ㄴ에서는 옆과 우에서 본 정투영의 보기체적을 보여 주며 ㄷ와 ㄹ에서는 옆과 우에서 본 비탈투영의 보기체적을 보여 준다.)

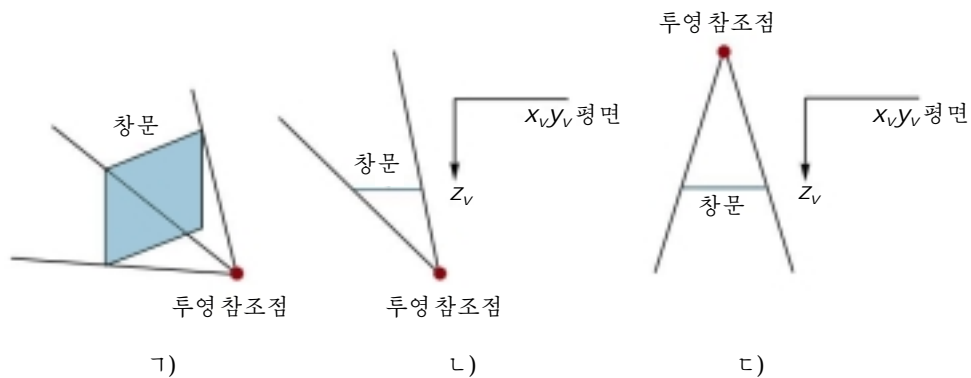



그림 12-29. 원근투영에서 투영참조점의 여러 위치에 대한 보기체적의 실례

유한보기체적은 체적의 범위를 z방향에서 제한하여 얻는다. 이것은 하나 또는 두개의 추가적인 경계면들의 위치를 지적하여 실현한다. 이 z_v경계면들을 보기체적의 앞면과 뒤면 또는 가까운 면과 먼면이라고 한다. 앞면과 뒤면은 지적된 위치 z_{front}와 z_{back}에서 보기면에 평행이다. 두 면은 다같이 투영참조점의 같은 쪽에 있어야 하며 뒤면은 앞면보다도 투영참조점으로부터 더 멀리 있어야 한다. 앞면과 뒤면을 포함하면 그림 12-30에 보여 준바와 같이 6개의 면들에 의하여 경계 지어지는 보기체적을 만든다. 6개의 면들은 정투영에서 직6면체모양의 보기체적을 만들며 한편 비탈투영에서는 빗평행6면체모양의 보기체적을 만든다. 원근투영에서 앞자르기면과 뒤자르기면은 무한각추를 잘라 각추대모양의 보기체적을 만든다.

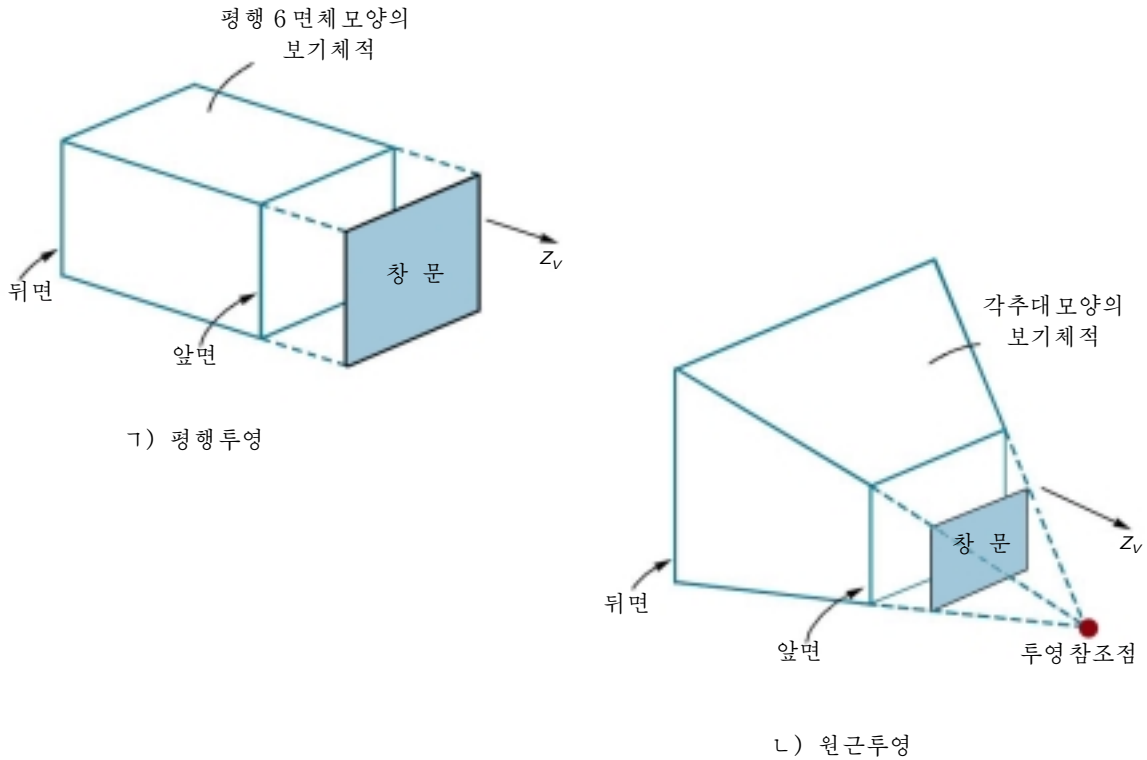



그림 12-30. 앞면과 뒤면, 웃면과 밑면, 옆면들로 경계 지어지는 보기체적(z_v 축우의 위치 z_{front}와 z_{back}에 있는 앞면과 뒤면은 보기면에 평행이다.)

보기조작에서 앞자르기면과 뒤자르기면은 깊이에 기초하여 장면의 부분들을 없앨수 있게 한다. 그러면 보기 좋은 장면의 부분들을 골라 낼수 있으며 보려고 하는 부분의 앞 또는 뒤에 있는 물체들을 없앨수 있다. 또한 원근투영인 경우 보기창문안에서 인식할수 없게 투영될수 있는 보기면가까이의 큰 물체들을 떼버리는데 앞자르기면을 리용할수 있다. 류사하게 뒤자르기면은 출력장치에 작은 점으로 투영될수 있는 투영참조점으로부터 먼 물체들을 잘라 버리는데 리용할수 있다.

보기면과 앞뒤자르기면의 상대적인 배치는 만들려는 보기의 류형과 개개 도형처리프로그람의 제한에 관계된다. PHIGS에서 보기면은 투영참조점을 포함하지 않는 한 z_v축우의 임의의 위치에 놓을수 있다. 그리고 앞뒤면은 투영참조점이 그 두 면사이에 있지 않는 한 보기면에 대하여 임의의 위치에 놓을수 있다. 그림 12-31에서는 보기면에 대하여 배치할수 있는 앞뒤면을 보여 주었다. PHIGS에서 기정의 보기체적은 $z_{front}=1$, $z_{back}=0$, 보기면은 뒤면과 일치하고 투영참조점은 앞면우의 위치(0.5, 0.5, 1.0)에 있는 평행투영을 리용하여 단위바른6면체로 형성된다.

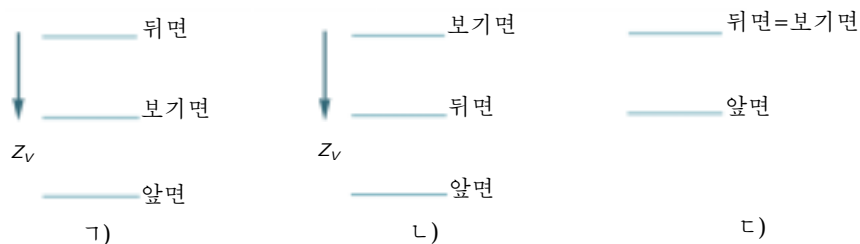



그림 12-31. 보기면에 대한 앞뒤자르기면의 가능한 배치

정투영은 보기면의 위치선정의 영향을 받지 않는다. 왜냐하면 투영선은 보기면의 위치에 무관계하게 거기에 수직이기때문이다. 그러나 비탈투영은 투영방향의 지적에 관계하는 보기면의 위치선정의 영향을 받을수 있다. PHIGS에서 비탈투영방향은 투영참조점으로부터 창문중심까지의 선에 평행이다. 따라서 투영참조점을 움직이지 않고 보기면의 위치를 움직이는것은 그림 12-32에 보여 준바와 같이 보기체적의 측면들의 비대칭도를 변화시킨다. 평행투영을 만들 때 보기면의 위치는 대체로 보기참조점 또는 앞자르기면에 정해 진다.

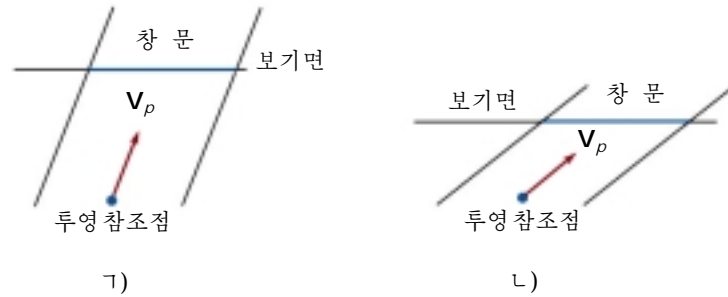



그림 12-32. 투영벡토르 $\mathbf{V}_p$가 투영참조점파 창문의 위치에 의하여 결정될 때 창문의 위치를 움직이는것에 의한 비탈투영의 보기체적의 련속형태변화

원근효과는 그림 12-33에 보여 준바와 같이 보기면에 대하여 투영참조점의 위치를 선정하는데 관계된다. 투영참조점을 보기면의 가까이에 놓으면 원근효과는 강조된다. 즉 보다 가까운 거리의 물체는 동일한 크기의 더 먼 거리의 물체보다 훨씬 크게 나타난다. 류사하게 투영참조점을 보기면으로부터 더 멀리로 움직일 때 멀고 가까운 물체들의 크기에서의 차이는 감소한다. 결국 투영참조점을 보기면으로부터 무한히 멀리로 움직일 때 원근투영은 평행투영에 접근한다.

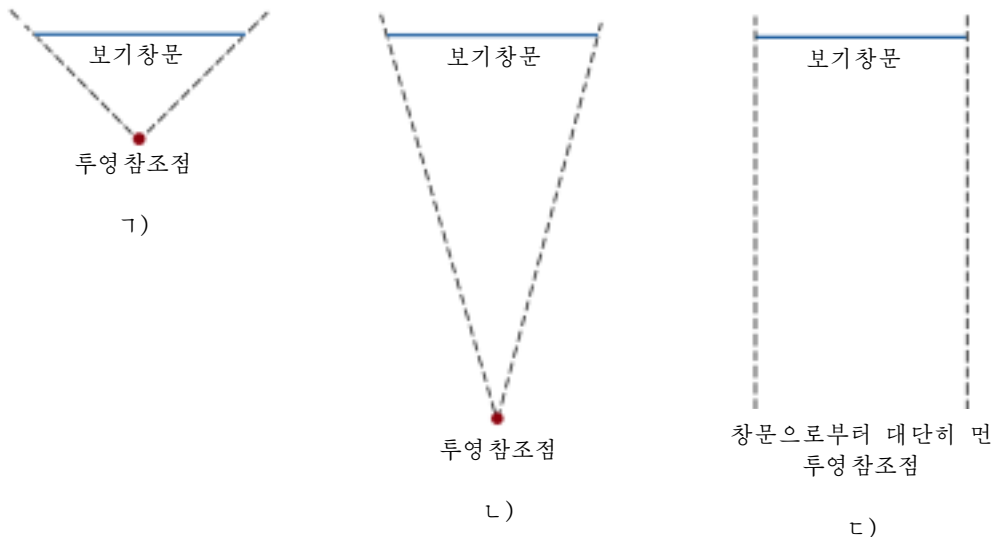



그림 12-33. 투영참조점을 보기면으로부터 멀리로 움직일 때 원근효과의 변화

원근보기에서 물체의 투영된 크기는 물체와 보기면의 상대적인 위치에 관계된다(그림 12-34). 보기면이 물체의 앞(투영참조점의 더 가까이)에 있으면 투영된 크기는 더 작다. 반대로 물체의 뒤에 있는 보기면에 투영할 때 물체의 크기는 커진다.

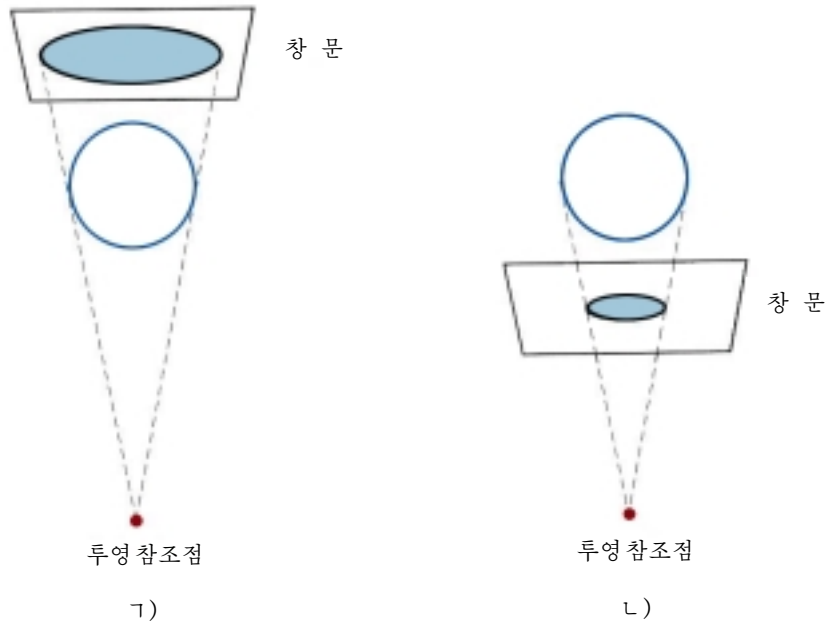



그림 12-34. 투영된 물체의 크기는 투영참조점의 위치에 대하여 보기면이 물체의 앞에 있는가 또는 뒤에 있는가에 관계된다.

원근투영에서 보기면의 위치선정은 장면을 정적으로 보려 하는가 아니면 동화를 만들려 하는가에 관계된다. 장면을 정적으로 볼 때 보기면은 보통 장면안의 편리한 점에 있는 보기자리표원점에 놓여 진다. 그러면 보려고 하는 장면의 모든 부분들을 포함하도록 창문의 크기를 조정하기가 쉽다. 투영참조점의 위치는 요구되는 원근량을 얻도록 정한다. 동화에서는 투영참조점을 보기자리표원점에 놓고 보기면을 장면의 앞에 놓을수 있다(그림 12-35). 이 배치는 카메라자리표계를 모의한다. 시야(렌즈각)는 투영참조점으로부터 보기면까지의 거리에 대하여 창문의 크기를 조정함으로써 설정한다. 보기자리표계를 움직이면 장면을 따라 가면서 볼수 있으며 투영참조점은 보기참조점과 함께 움직일것이다.

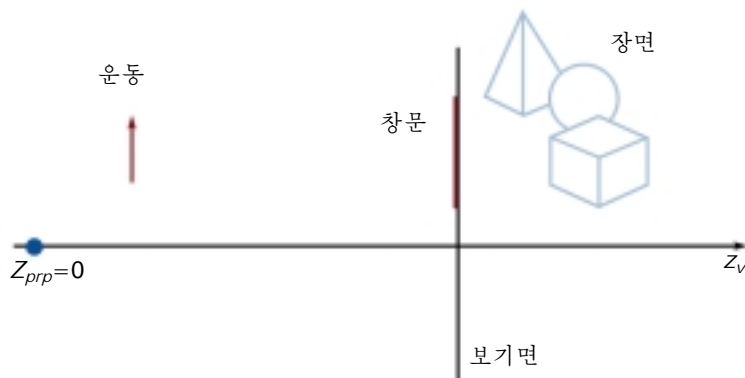



그림 12-35. 동화에서 카메라자리표계를 모의하기 위한 보기면의 위치선정

일반평행투영변환

PHIGS에서 평행투영의 방향은 투영참조점으로부터 보기창문의 중심에로의 투영벡토르에 의하여 지적된다. 그림 12-36은 주어 진 투영벡토르와 보기면의 투영창문에 대한 유한보기체적의 일반적인

형태를 보여 준다. 비탈투영변환은 그림 12-36의 보기체적을 그림 12-37에 보여 준 직6면체로 변환하는 쏠림조작에 의하여 얻는다.

그림 12-37에 보여 준 보기체적을 만드는데 필요한 쏠림변환의 원소들은 투영벡토르의 쏠림변환을 고찰하여 얻는다. 투영벡토르가 세계자리표로 지적되면 그것을 먼저 12장 2절에서 설명한 회전행렬을 리용하여 보기자리표로 변환하여야 한다(투영벡토르는 평행이동의 영향을 받지 않는다. 왜냐하면 그것은 단순히 고정된 위치가 없는 방향이기때문이다.). 투영벡토르를 보기자리표로 지적하는 도형처리프로그람들에서는 투영벡토르의 입력원소들에 쏠림을 직접 적용한다.

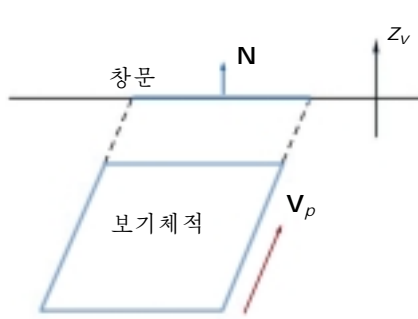



그림 12-36. 비탈투영벡토르와 보기체적

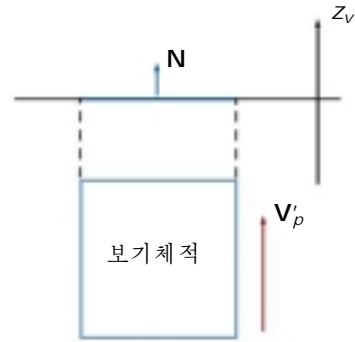



그림 12-37. 그림 12-36의 보기체적을 쏠리게 하여 얻는 직6면체 모양의 보기체적

투영벡토르의 원소들이 보기자리표로

$$\mathbf{V}_p = (p_x, p_y, p_z) \tag{12-19}$$

라고 하자. 투영벡토르 $\mathbf{V}_p$를 보기면의 법선벡토르 $\mathbf{N}$과 정렬시키는 쏠림행렬의 원소들을 결정하여야 한다(그림 12-37). 이 변환은

$$\mathbf{V}'_p = \mathbf{M}_{\text{parallel}} \cdot \mathbf{V}_p = \begin{bmatrix} 0 \\ 0 \\ p_z \\ 0 \end{bmatrix} \tag{12-20}$$

과 같이 표현할수 있다. 여기서 $\mathbf{M}_{\text{parallel}}$은 평행투영행렬 12-10과 등가이며

$$\mathbf{M}_{\text{parallel}} = \begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12-21}$$

의 형식의 z축쏠림을 표현한다. 식 12-20으로부터 쏠림파라메터 a와 b에 의한 양함수적인 변환방정식은

$$\begin{aligned} 0 &= p_x + ap_z \\ 0 &= p_y + bp_z \end{aligned} \tag{12-22}$$

이며 쏠림파라메터들의 값은 다음과 같다.

$$a = -\frac{p_x}{p_z}, \qquad b = -\frac{p_y}{p_z} \tag{12 - 23}$$

따라서 투영벡토르의 원소들에 의하여

$$\mathbf{M}_{\text{parallel}} = \begin{bmatrix} 1 & 0 & -p_x/p_z & 0 \\ 0 & 1 & -p_y/p_z & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12-24)$$

와 같은 일반평행투영행렬을 가진다. 이 행렬은 다음에 세계자리표로부터 평행투영자리표에로의 변환을 만들기 위하여 12장 2절의 변환 **R·T**와 결합된다. 정투영에서 $p_x = p_y = 0$이며 $\mathbf{M}_{\text{parallel}}$은 단위행렬이다. 그림 12-38로부터 투영벡토르의 성분들을 파라메터 L, α 및 ϕ(12장 3절)에 관련시킬수 있다. 닮음3각형들에 의하여

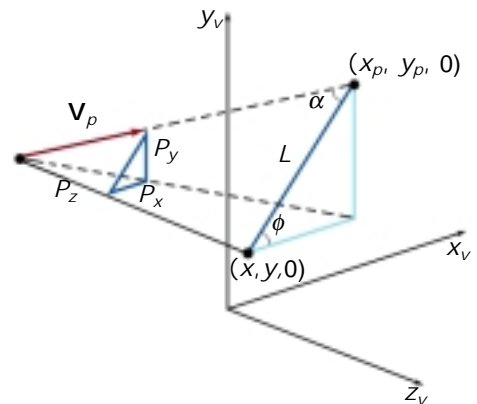
$$\frac{L\cos\phi}{z} = -\frac{p_x}{p_z}$$
$$\frac{L\sin\phi}{z} = -\frac{p_y}{p_z} \qquad (12-25)$$

이다. 이것은 변환행렬식 12-10과 12-24의 원소들이 동등함을 설명한다. 식 12-25에서 z와 p_z는 부호가 반대이며 그림 12-38에서 보여 주는 위치들에 대하여 $z<0$이다.

일반원근투영변환

PHIGS프로그람작성표준에서 투영참조점은 보기면 또는 앞뒤자르기면사이를 제외하고 보기자리표계의 임의의 위치에 선정될수 있다. 그림 12-39는 투영참조점의 임의의 위치에 대한 유한보기체적의 형태를 보여 준다. 일반원근투영변환을 다음의 두 조작에 의하여 얻을수 있다.

그림 12-38. 평행투영벡토르 $\mathbf{V}_p$와 파라메터 L, α, ϕ사이의 관계

1. 각추대의 중심선이 보기면에 수직이 되도록 보기체적을 쏠리게 한다.
2. $1/z$에 관계되는 비례결수에 의하여 보기체적을 비례변환한다.

두번째 걸음(보기체적의 비례변환)은 12장 3절에서 설명한 원근변환과 등가이다.

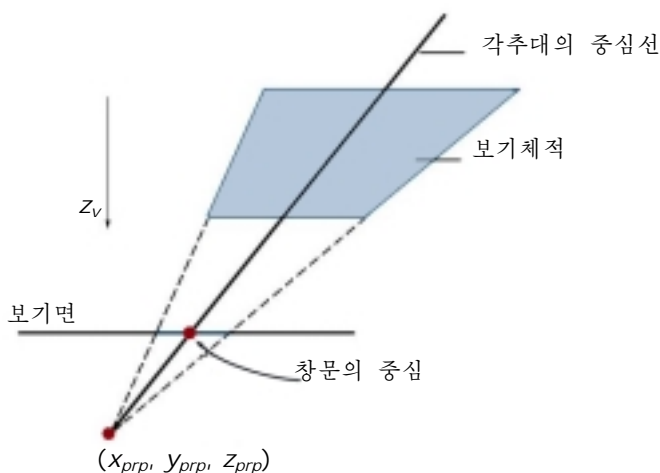



그림 12-39. z_v축에 있지 않는 투영참조점을 가지는 원근보기체적의 일반형태

일반적인 원근보기체적을 투영창문에 정렬시키는 쏠림조작을 그림 12-40에 보여 주었다. 이 변환은 창문의 중심을 포함하여 각추대의 중심선우의 모든 위치들을 보기면에 수직인 선에로 옮기는 효과를 가진다. 일반위치 $(x_{prp}, y_{prp}, z_{prp})$의 투영참조점에 대하여서는 z축쏠림과 평행이동을 결합하여 이 변환을 수행한다.

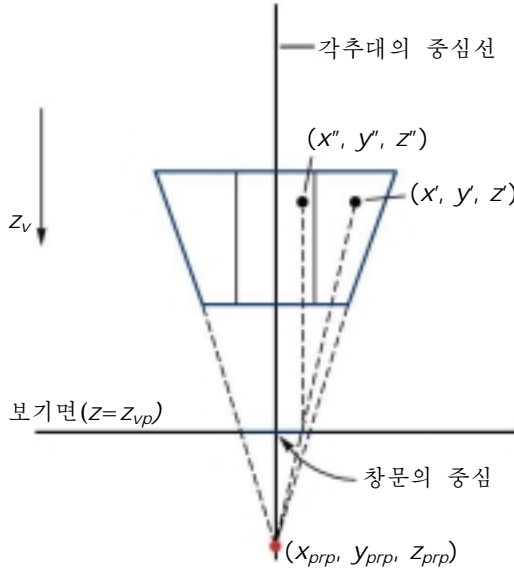



그림 12-40. 일반원근보기체적을 투영창문과 중심맞추는 쏠림

$$\mathbf{M}_{\text{parallel}} = \begin{bmatrix} 1 & 0 & a & -az_{prp} \\ 0 & 1 & b & -bz_{prp} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (12-26)$$

여기서 쏠림파라메터는

$$a = -\frac{x_{prp} - (xw_{\min} + xw_{\max})/2}{z_{prp}}$$
$$b = -\frac{y_{prp} - (yw_{\min} + yw_{\max})/2}{z_{prp}} \quad (12-27)$$

이다. 보기체적안의 점들은 이 조작에 의하여

$$x' = x + a(z - z_{prp})$$
$$y' = y + b(z - z_{prp}) \quad (12-28)$$
$$z' = z$$

와 같이 변환된다. 투영참조점이 z_v축에 있을 때 $x_{prp}=y_{prp}=0$이다.

본래의 보기체적안의 위치(x, y, z)를 쏠려 진 각추대안의 위치 (x', y', z')로 변환하였다면 정규직6면체를 만드는데 비례변환을 적용한다(그림 12-40). 이 변환은

$$x'' = x'\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + x_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right)$$
$$y'' = y'\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + y_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right) \quad (12-29)$$

이며 동차행렬표현은

$$\mathbf{M}_{\text{scale}} = \begin{bmatrix} 1 & 0 & \dfrac{-x_{prp}}{z_{prp}-z_{vp}} & \dfrac{x_{prp}z_{vp}}{z_{prp}-z_{vp}} \\ 0 & 1 & \dfrac{-y_{prp}}{z_{prp}-z_{vp}} & \dfrac{y_{prp}z_{vp}}{z_{prp}-z_{vp}} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \dfrac{-1}{z_{prp}-z_{vp}} & \dfrac{z_{prp}}{z_{prp}-z_{vp}} \end{bmatrix} \qquad (12-30)$$

이다.

따라서 일반원근투영변환은 행렬형식으로

$$\mathbf{M}_{\text{perspective}} = \mathbf{M}_{\text{scale}}\mathbf{M}_{\text{shear}} \qquad (12-31)$$

와 같이 표현할수 있다. 세계자리표로부터 원근투영자리표에로의 완전한 변환은 $\mathbf{M}_{\text{perspective}}$의 오른쪽에 12장 2절의 합성보기변환 $\mathbf{R}\cdot\mathbf{T}$를 련결하여 얻는다.

5절. 자르기

이 절에서는 먼저 보기체적의 자르기면들을 직접 리용하여 어떻게 자르기할수 있는가를 고찰함으로써 3차원자르기의 일반적인 방법들을 연구한다. 다음에 정규화된 보기체적과 동차자리표를 리용하는 보다 효률적인 방법들을 론의한다.

3차원자르기알고리듬은 보기체적안의 모든 부분면들을 출력장치에 현시하기 위하여 그것들을 식별하고 보관한다. 보기체적의 바깥쪽에 있는 물체의 모든 부분들은 버린다. 3차원자르기는 2차원자르기방법을 확장하여 수행할수 있다. 창문의 경계선에 대한 자르기대신에 보기체적의 경계면에 대하여 물체를 자른다.

선분을 보기체적으로 자르자면 보기체적의 경계면의 방정식을 리용하여 선분의 상대위치를 검사하여야 한다. 선분의 끝점들의 자리표를 매개 경계면의 방정식에 차례로 대입하여 끝점이 그 경계면의 안쪽 또는 바깥쪽에 있는가를 결정할수 있다. 선분의 끝점 (x, y, z)는 $Ax+By+Cz+D>0$이면 경계면의 바깥쪽에 있다. 여기서 A, B, C, D는 경계면의 파라메터들이다. 류사하게 $Ax+By+Cz+D<0$이면 점은 경계면의 안쪽에 있다. 두 끝점이 다 경계면의 바깥쪽에 있는 선분은 버리고 두 끝점이 다 모든 경계면들의 안쪽에 있는 선분은 보관한다. 선분과 경계면과의 사귐점은 직선의 방정식과 평면의 방정식을 함께 리용하여 구한다. 사귐점의 자리표 (x_1, y_1, z_1)는 그 직선우에 있으며 평면의 방정식 $Ax_1+By_1+Cz_1+D=0$을 만족하는 값들이다.

다각형을 자르기 위하여 다각형의 개별적인 변들을 자를수 있다. 우선 보기체적의 매 경계면에 대하여 다각형이 그 경계면의 완전한 내부 또는 완전한 외부인가를 결정하기 위하여 다각형의 자리표범위를 검사할수 있다. 다각형의 자리표범위가 모든 경계면들의 내부이면 다각형을 보관하고 자리표범위가 모든 경계면들의 외부이면 다각형을 버린다. 그렇지 않으면 사귐점계산을 적용할 필요가 있다. 이것은 앞단락에서 설명한바와 같이 보기체적의 경계면과 다각형의 변과의 사귐점위치를 결정하여 할수 있다.

2차원보기에서와 마찬가지로 보기체적에 의한 자르기전 또는 자르기후에 투영조작을 할수 있다. 보기체적안의 모든 물체들은 지적된 투영창문의 내부에로 넘어 간다. 마지막걸음은 창문의 내용을 출력장치에서의 현시위치를 지적하는 2차원보임창에로 변환하는것이다.

2차원에서 자르기는 일반적으로 곧바로 선 직4각형에 대하여 수행된다. 즉 자르기창문은 x 및 y

축에 관하여 정렬된다. 이것은 자르기계산을 크게 간단화한다. 왜냐하면 창문의 매 경계선은 하나의 자리표값에 의하여 정의되기때문이다. 실례로 창문의 왼쪽 경계선을 지나는 모든 선들의 사귐점은 그 왼쪽 경계선과 같은 x자리표를 가진다.

보기체적의 자르기경계면의 방향은 투영의 류형, 투영창문, 투영참조점의 위치에 관계된다. 앞자르기면과 뒤자르기면은 보기면에 평행이기때문에 매면은 다 상수 z자리표값을 가진다. 선과 이 면들과의 사귐점의 z자리표는 간단히 대응하는 면의 z자리표이다. 그러나 보기체적의 다른 4개의 옆면들은 임의의 공간적인 방향을 가질수 있다. 선과 보기체적의 한 경계면과의 사귐점을 찾아내기 위해서는 그 경계면의 방정식을 얻어야 한다. 이 처리는 자르기전에 보기체적을 직6면체로 변환하면 간단화된다. 다시 말하여 먼저 보기체적안의 자리표값들을 직각평행자리표로 변환하는 투영변환을 수행하고 다음에 자르기계산을 수행한다.

정규직6면체에 대한 자르기는 매개 면이 자리표축들중 하나에 수직이기때문에 훨씬 더 쉽다. 그림 12-41에서 보는바와 같이 보기체적의 웃면과 밑면은 상수 y의 면이며 옆면들은 상수 x의 면, 앞면과 뒤면은 상수 z의 면이다. 실례로 직6면체의 웃면을 지나는 선은 y자리표값이 웃면의 y자리표와 같은 사귐점을 가진다.

그림 12-41. 직6면체모양의 보기체적과 사귀는 물체

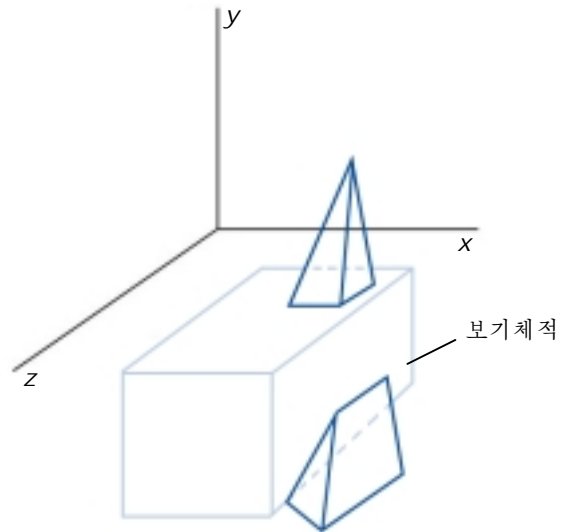
정투영인 경우 보기체적은 이미 직6면체이다. 12장 3절에서 본바와 같이 비탈투영의 보기체적은 쏠림조작에 의하여 직6면체에로 변환되며 원근보기체적은 일반적으로 쏠림-비례변환결합에 의해 변환된다.

정규화된 보기체적

그림 12-42는 확장된 PHIGS의 변환경로를 보여 준다. 첫번째 걸음에서 장면의 물체표현은 모형화자리표표현으로부터 세계자리표표현으로 변환된다. 다음에 보기변환은 세계자리표를 보기자리표로 변환한다. 투영단계에서 보기자리표는 투영자리표로 변환되며 이때 보기체적은 직6면체에로 효과적으로 변환된다. 다음에 직6면체는 단위바른6면체 즉 정규화된 투영자리표계라고 부르는 정규화된 보기체적에로 넘어 간다. 정규화된 투영자리표에로의 넘기기는 직6면체안의 점들을 단위바른6면체의 부분 또는 전부를 차지하는 지적된 3차원보임창안의 위치에로 변환하여 수행한다. 마지막으로 워크스테이션단계에서 정규화된 투영자리표는 현시하는 장치의 자리표에로 변환된다.

정규화된 보기체적은 평면

$$x=0,\quad x=1,\quad y=0,\quad y=1,\quad z=0,\quad z=1 \qquad (12-32)$$

에 의하여 정의되는 공간이다. 류사한 변환렬은 다른 도형처리프로그람들에서 체계에 의존하는 개별적인 변화들과 함께 리용된다. 실례로 GL프로그람은 직6면체를 매개 자리표방향에서 ±1위치에 경계면이 있는 바른6면체의 내부에로 넘긴다.

본래의 보기체적 또는 지어 투영자리표계에서의 직6면체대신에 단위바른6면체로 자르는것은 여러가지 우점이 있다. 첫째로 정규화된 보기체적은 임의의 크기의 보기체적에 대하여 표준적인 표현을 준다. 이것은 보기변환을 임의의 워크스테이션고찰로부터 분리시키며 단위바른6면체는 다음에 임의의

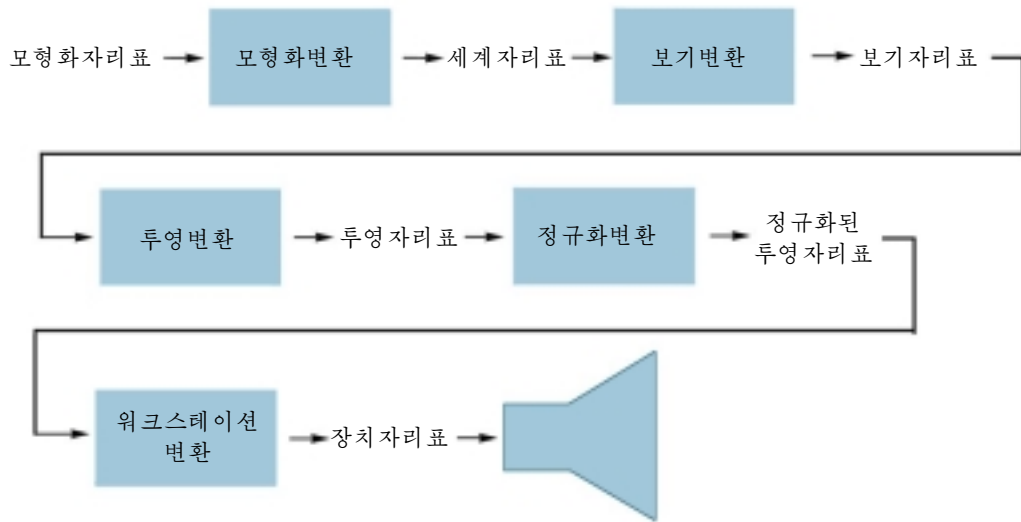



그림 12-42. 확장된 PHIGS 의 변환경로

크기로 워크스테이션에 넘겨 질수 있다. 둘째로 자르기절차들은 단위자르기면 또는 보임창면에 의하여 간단화 및 표준화되며 장치자리표에로의 변환전에 정규공간에서 추가적인 자르기면들을 지적할수 있다. 셋째로 깊이삽입과 보이는 면의 결정은 간단화된다. 왜냐하면 z축은 항상 보는 사람쪽을 가리키기때문이다(투영참조점은 지금 z축우의 점으로 변환되였다.). 물체의 앞면은 z방향성분이 정인 법선벡토르를 가지는 면이며 뒤면은 부의 z방향에 면한다.

직각보기체적안의 위치를 3차원의 직각보임창에로 넘기는것은 2차원의 창문-보임창넘기기에 필요한 조작과 류사한 비례변환 및 평행이동의 결합에 의하여 수행된다. 이 조작에 대한 3차원변환행렬을

$$\begin{bmatrix} D_x & 0 & 0 & K_x \\ 0 & D_y & 0 & K_y \\ 0 & 0 & D_z & K_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (12-33)$$

의 형식으로 표현할수 있다. 인자 D_x, D_y, D_z는 보임창과 직6면체모양의 보기체적의 x, y, z방향크기의 비이다(그림 12-43).

$$D_x = \frac{xv_{\max} - xv_{\min}}{xw_{\max} - xw_{\min}}$$
$$D_y = \frac{yv_{\max} - yv_{\min}}{yw_{\max} - yw_{\min}} \qquad (12-34)$$
$$D_z = \frac{zv_{\max} - zv_{\min}}{z_{back} - z_{front}}$$

여기서 보기체적의 경계는 창문경계 ($xw_{\min}$, $xw_{\max}$, $yw_{\min}$, $yw_{\max}$)와 앞뒤면의 위치 z_{front}와 z_{back}에 의하여 설정된다. 보임창의 경계는 자리표값 $xv_{\min}$, $xv_{\max}$, $yv_{\min}$, $yv_{\max}$, $zv_{\min}$, $zv_{\max}$에 의하여 설정된다. 변환에서 더하는 평행이동인자 K_x, K_y, K_z는

$$K_x = xv_{min} - xw_{min} D_x$$
$$K_y = yv_{min} - yw_{min} D_y \qquad (12-35)$$
$$K_z = zv_{min} - z_{front} D_z$$

이다.

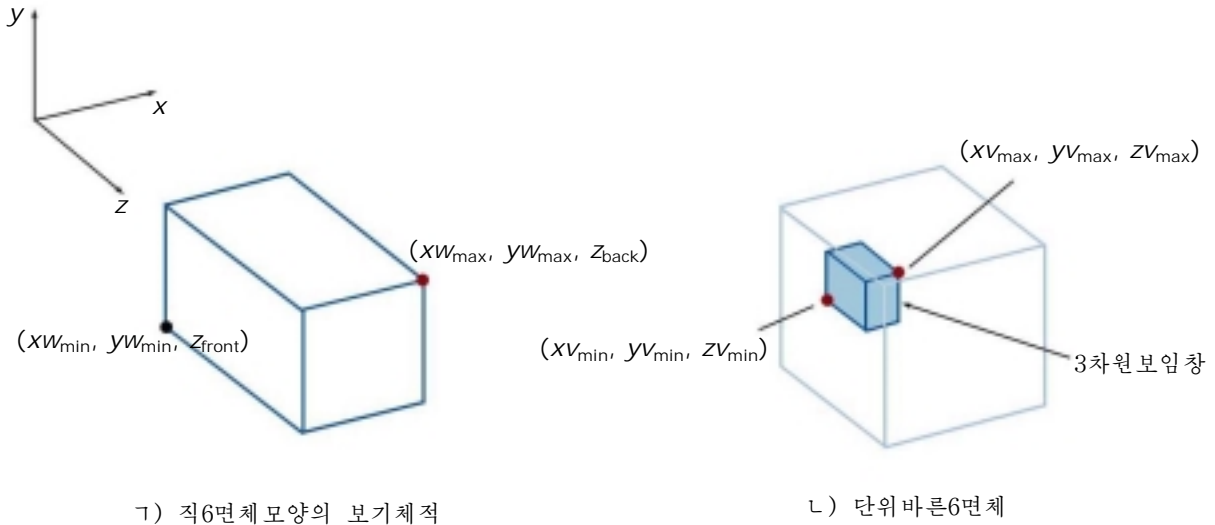



그림 12-43. 보기체적과 3차원보임창의 크기

보임창자르기

장면안의 선과 다각형들은 물체가 2차원보임창의 변대신에 3차원보임창의 자르기면에 대하여 처리된다는것을 제외하고 2차원의것과 류사한 절차에 의하여 3차원보임창의 경계면들로 잘라 질수 있다. 곡면은 그의 방정식을 리용하여 평행6면체의 면들과의 사귐선들의 위치를 정함으로써 처리된다.

2차원구역코드의 개념은 체적의 왼쪽, 오른쪽, 우, 아래의 위치뿐아니라 3차원보임창의 앞과 뒤의 위치를 고려하여 3차원에로 확장할수 있다. 2차원자르기에서 보임창의 경계선에 대한 선분의 끝점들의 위치를 식별하기 위하여 4자리 2진구역코드를 리용하였다. 3차원의 점에 대하여 구역코드를 6bit로 확장하여야 한다. 그러면 장면표현안의 매점에는 보임창에 대한 그점의 상대위치를 식별하는 6bit 구역코드가 할당된다. 위치가 (x, y, z)인 선분의 한 끝점에 대한 구역코드에서 비트위치는 오른쪽에서 왼쪽으로

비트 $1=1, \quad x < xv_{min}$ (왼쪽)
비트 $2=1, \quad x > xv_{max}$ (오른쪽)
비트 $3=1, \quad y < yv_{min}$ (아래)
비트 $4=1, \quad y > yv_{max}$ (우)
비트 $5=1, \quad z < zv_{min}$ (앞)
비트 $6=1, \quad z > zv_{max}$ (뒤)

와 같이 할당된다. 실례로 구역코드 101000은 보임창의 웃쪽 뒤의 점을 지적하며 구역코드 000000은 체적내부의 점을 지적한다.

선분은 두 끝점이 다 구역코드 000000을 가지면 완전히 보임창내부에 있다고 인차 식별할수 있다. 선분의 한 끝점이 구역코드 000000을 가지지 않는다면 두 끝점의 구역코드들에 대하여 론리곱하기연산을 수행한다. 이 론리곱하기연산의 결과는 두 끝점이 다 동일한 바깥구역에 있는 임의의 선분에 대하여 0이 아닐것이다. 실례로 0아닌 값은 두 끝점이 다 보임창의 뒤에 있거나 또는 보임창의 우에 있으면 발생될것이다. 선분을 체적의 완전한 내부 또는 완전한 외부라고 식별할수 없으면 체적의 경계면과의 사귐점에 대해 시험한다.

2차원의 선분자르기에서와 마찬가지로 선분의 얼마만한 부분을 버릴수 있는가를 결정하기 위하여 선분과 보임창의 면과의 계산된 사귐점을 리용한다. 선분의 나머지 부분은 보임창의 다른 면들에 대하여 검사되며 선분이 총체적으로 버려지거나 또는 그 부분이 체적의 내부라는것이 발견될 때까지 검사를 계속한다.

3차원선분의 방정식은 보조변수형식으로 편리하게 표현된다. 시루스-벡크 또는 리앙-바스키의 2차원보조변수자르기방법은 3차원장면에로 확장할수 있다. 끝점 $\mathbf{P}_1=(x_1, y_1, z_1)$와 $\mathbf{P}_2=(x_2, y_2, z_2)$을 가지는 선분에 대하여 선분의 보조변수방정식을

$$\begin{aligned} x &= x_1+(x_2-x_1)u, \qquad 0\le u\le 1 \\ y &= y_1+(y_2-y_1)u \\ z &= z_1+(z_2-z_1)u \end{aligned} \tag{12-36}$$

와 같이 쓸수 있다. 자리표 (x, y, z)는 선분의 두 끝점사이의 임의의 점을 표현한다. $u=0$에서 점 $\mathbf{P}_1$이고 $u=1$에서 점 $\mathbf{P}_2$이다.

선분과 보임창의 면과의 사귐점을 찾기 위하여 그 면의 자리표값을 식 12-36의 적당한 보조변수식에 대입하고 u에 대하여 푼다. 실례로 선분을 보임창의 zv_{min}면에 대하여 검사한다고 하자. 그러면

$$u=\frac{zv_{min}-z_1}{z_2-z_1} \tag{12-37}$$

이다. u의 계산된 값이 0~1사이의 범위에 있지 않을 때 선분은 끝점 $\mathbf{P}_1$와 $\mathbf{P}_2$사이의 그 어느 점에서도 면과 사귀지 않는다(그림 12-44의 선분 A). 식 12-37에서 u의 계산된 값이 0~1사이의 구간에 있으면 사귐점의 x 및 y 자리표를

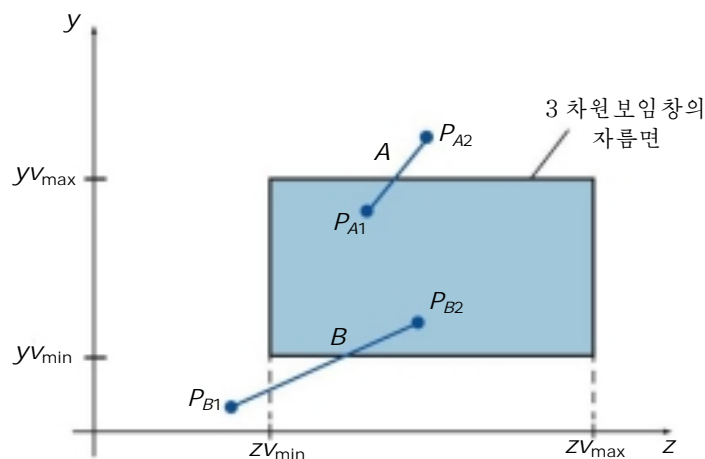



그림 12-44. 보임창의 zv_{min}면에 대하여 잘라 질 두 선분을 옆에서 보기(선분 A에 대하여 식 12-37은 0~1사이의 범위밖에 있는 u의 값을 만든다. 선분 B에 대하여 식 12-38은 yv_{min}부터 yv_{max}까지의 범위밖에 있는 사귐점의 자리표를 만든다.)

$$x_1 = x_1 + (x_2 - x_1)\left(\frac{zv_{\min} - z_1}{z_2 - z_1}\right)$$
$$y_1 = y_1 + (y_2 - y_1)\left(\frac{zv_{\min} - z_1}{z_2 - z_1}\right) \qquad (12-38)$$

과 같이 계산한다. x_1 또는 y_1가 보임창의 경계범위에 있지 않으면 이 선분은 체적의 앞경계면의 밖에서 앞면과 사귄다(그림 12-44의 선분 B).

동차자리표에 의한 자르기

3차원자리표를 리용하여 자르기절차를 설명하였지만 PHIGS 및 기타 프로그람들은 실제로 자리표위치를 동차자리표로 표현한다. 이것은 여러가지 변환들이 4×4행렬로 표현될수 있게 하며 그것들은 효률적으로 련결될수 있다. 모든 보기 및 기타 변환들이 수행된후 동차자리표위치는 3차원의 점에로 다시 변환된다.

매개 자리표위치가 변환경로에 들어 갈 때 그것은 동차자리표표현으로 변환된다.

$$(x, y, z) \rightarrow (x, y, z, 1)$$

여러가지 변환들이 적용되고 마지막동차점

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (12-39)$$

을 얻는다. 여기서 동차파라메터 h는 1이 아닐수 있다. 사실상 h는 임의의 실수값을 가질수 있다. 자르기는 다음에 동차자리표에 의하여 수행되며 잘라 진 동차위치들은 3차원정규투영자리표계에서의 비동차자리표로 변환된다.

$$x' = \frac{x_h}{h}, \qquad y' = \frac{y_h}{h}, \qquad z' = \frac{z_h}{h} \qquad (12-40)$$

물론 파라메터 h의 크기가 대단히 작거나 또는 값 0을 가지면 문제가 생길것이다. 그러나 변환들이 적당히 수행되면 이것은 표준적으로는 일어 나지 않을것이다. 변환경로의 마지막단계에서 정규화된 점은 3차원의 장치자리표점으로 변환된다. xy위치는 장치에 표시되며 z성분은 깊이정보처리에 리용된다.

동차자리표에 의하여 자르기절차를 수행하는것은 하드웨어적으로 보기를 실현할 때 평행 및 원근투영변환에 대하여 다같이 단일한 절차를 리용할수 있게 한다. 평행투영되는 물체는 3차원정규자리표에 의하여 정확히 잘라 질것이며 값 h=1은 다른 조작들에 의하여 변경되지 않는다. 그러나 원근투영은 일반적으로 더이상 값 1을 가지지 않는 동차파라메터를 만든다. 쏠려 진 각추대를 직6면체로 변환하는것은 동차파라메터의 값을 변화시킬수 있다. 그러므로 자르기가 정확히 수행된다는것을 확신하기 위하여 동차자리표로 자르기하여야 한다. 또한 유리스플라인표현은 h<1을 포함하여 임의의 값의 동차파라메터를 가지는 동차자리표들로 설정된다. 부의 값 동차파라메터는 또한 자리표위치가 투영참조점의 뒤에 있는 원근투영에서 발생될수 있다. 이것은 건물이나 다른 물체의 내부를 보기 위하여 그것들의 내부에서 움직이려 하는 응용에서 일어 날수 있다.

동차보임창자르기경계를 결정하기 위하여 임의의 동차자리표위치 (x_h, y_h, z_h, h)는 부등식

$$xv_{\min} \le \frac{x_h}{h} \le xv_{\max}, \qquad yv_{\min} \le \frac{y_h}{h} \le yv_{\max}, \qquad zv_{\min} \le \frac{z_h}{h} \le zv_{\max} \qquad (12-41)$$

을 만족하면 보임창내부에 있다는것을 언급한다. 이리하여 동차자르기경계는

$$\begin{aligned} &hxv_{\min} \le x_h \le hxv_{\max}, \quad hyv_{\min} \le y_h \le hyv_{\max}, \quad hzv_{\min} \le z_h \le hzv_{\max}, \quad h > 0 \\ &hxv_{\max} \le x_h \le hxv_{\min}, \quad hyv_{\max} \le y_h \le hyv_{\min}, \quad hzv_{\max} \le z_h \le hzv_{\min}, \quad h < 0 \end{aligned} \qquad (12-42)$$

이다. 그리고 자르기는 앞부분에서 설명한것과 류사한 절차로 수행된다. 식 12-42의 부등식들을 다같이 적용하는것을 피하기 위하여 $h<0$인 임의의 점에 대하여 자리표들의 부호를 간단히 반대로 하고 $h>0$에 대한 자르기부등식을 리용할수 있다.

6절. 하드웨어실현

대부분의 도형처리과정들은 지금 하드웨어적으로 실현된다. 대표적으로 보기, 보이는 면의 식별, 명암알고리듬들은 대규모집적회로(VLSI)기술에 기초하는 도형처리소편들을 리용하여 수행할수 있다. 3차원 또는 2차원의 응용을 위하여 물체를 변환하고 자르며 출력장치에로 투영하는 하드웨어체계들이 설계되고 있다.

그림 12-45는 이 장에서 설명한 보기조작들을 수행하는 도형처리소편모임에서 요소들의 배렬을 설명한다. 소편들은 기하학적인 변환, 자리표계의 변환, 투영 및 자르기를 수행하기 위한 경로로 조직된다. 첫 4개의 소편들은 비례변환, 평행이동, 회전, 세계자리표를 투영자리표로 변환하는데 필요한 변환들을 수행하는 행렬연산들을 위하여 제공된다. 다음의 6개 소편들은 보임창의 한 경계면에 하나씩 대응하여 자르기를 수행한다. 이 소편들중 4개는 2차원응용에 리용되며 다른 2개는 3차원보임창의 앞면과 뒤면에 대한 자르기에 필요하다. 경로에서 마지막 2개 소편은 보임창자리표를 출력장치자리표로 변환한다. 완전한 3차원도형처리체계를 제공하기 위하여 보이는 면의 식별과 면명암알고리듬을 수행하는 요소들을 이 모임에 추가할수 있다.

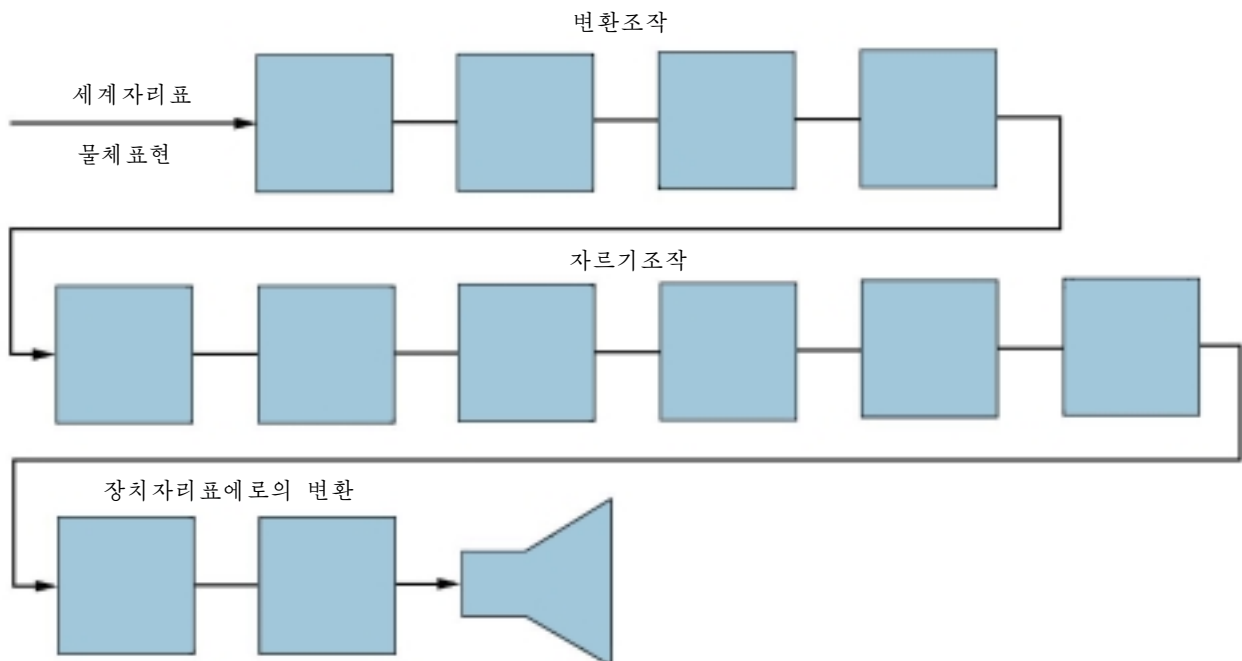



그림 12-45. 자리표변환과 자르기조작에 대하여 12개의 소편을 리용하는 3차원보기조작의 하드웨어실현

다른 전문화된 하드웨어체계들이 개발되였다. 여기에는 8분나무표현처리 및 광선추적알고리듬을 리용하여 3차원장면을 현시하는 하드웨어체계들이 있다(14장).

7절. 3차원보기함수

응용프로그람에서 보기변환에 대한 파라메터들을 설정할수 있도록 3차원도형처리서고에 여러가지 절차들을 제공한다. 이 절차들을 만드는데는 여러가지 방법들이 있지만 여기서는 3차원보기에 대한 PHIGS함수들을 설명한다.

파라메터들이 세계자리표로 지적될 때 세계자리표를 보기자리표로 변환하는 행렬의 원소들은 함수

```
evaluateViewOrientationMatrix3 (x0, y0, z0, xN, yN, zN,
                               xV, yV, zV, error, viewMatrix)
```

를 리용하여 계산한다. 이 함수는 12장 2절에서 설명한바와 같이 보기자리표계를 정의하는 입력자리표들로부터 viewMatrix를 만든다. 파라메터 x0, y0, z0은 보기자리표계의 원점(보기참조점)을 지적한다. 세계자리표벡토르 (xN, yN, zN)은 보기면의 법선과 보기자리표계의 정의 z_v축의 방향을 정의한다. 그리고 세계자리표벡토르 (xV, yV, zV)는 보기의 웃방향벡토르의 원소들을 준다. (xN, yN, zN)에 수직인 이 벡토르의 투영은 보기자리표계의 정의 y_v축의 방향을 설정한다. 입력값들이 정확히 지적되지 않으면 파라메터 error에 옹근수오유코드가 발생된다. 실례로 (xV, yV, zV)를 (xN, yN, zN)에 평행으로 설정하면 오유가 발생될것이다.

다른 보기자리표계를 지적하기 위하여 자리표파라메터들의 일부 또는 전체를 재정의하고 새로운 행렬을 지적하여 evaluateViewOrientationMatrix3을 실시할수 있다. 이 방법으로 임의의 개수의 세계-보기자리표변환행렬들을 설정할수 있다.

보기자리표를 정규화된 투영자리표로 변환하는 행렬 projMatrix는 함수

```
evaluateViewMappingMatrix3 (xwmin, xwmax, ywmin, ywmax,
          xvmin, xvmax, yvmin, yvmax, zvmin, zvmax,
          projType, xprojRef, yprojRef, zprojRef, zview,
          zback, zfront, error, projMatrix)
```

에 의하여 만들어 진다. 보기면에서 창문의 경계들은 보기자리표로 파라메터 xwmin, xwmax, ywmin, ywmax에 주어 진다. 단위바른6면체안의 3차원보임창의 경계들은 정규화된 자리표 xvmin, xvmax, yvmin, yvmax, zvmin, zvmax에 의하여 설정된다. 파라메터 projType는 투영의 류형을 *parallel* 또는 *perspective*로 선택하는데 리용된다. 자리표위치 (xprojRef, yprojRef, zprojRef)는 투영참조점을 설정한다. 이 점은 projType가 *perspective*로 설정되면 투영의 중심으로 리용되며 그렇지 않으면 이 점과 보기면의 창문의 중심은 평행투영벡토르를 정의한다. 보기자리표계의 z_v축을 따라 보기면의 위치는 파라메터 zview에 의하여 설정된다. z_v축을 따라 보기체적의 앞면과 뒤면의 위치는 파라메터 zfront와 zback에 의하여 주어 진다. 그리고 error파라메터는 틀린 입력자료를 지적하는 옹근수오유코드를 준다. 여러가지 3차원의 보기와 투영을 얻기 위하여 임의의 개수의 투영변환행렬들을 이 함수에 의하여 만들수 있다.

보기행렬과 투영행렬은 지적된 워크스테이션에서

```
setViewRepresentation3 (ws, viewIndex, viewMatrix, projMatrix,
          xclipmin, xclipmax, yclipmin, yclipmax, zclipmin,
          zclipmax, clipxy, clipback, clipfront)
```

에 의하여 결합할수 있다. 파라메터 ws는 워크스테이션을 선택하는데 리용되며 파라메터 viewMatrix와 projMatrix는 결합할 보기행렬과 투영행렬이다. 이 행렬들의 결합은 워크스테이션의 보기표에 넣어 지며 파라메터 viewIndex에 할당된 옹근수값에 의하여 참조된다. 장면을 자르기 위하여 정규투영자리표로 주어 지는 경계들은 파라메터 xclipmin, xclipmax, yclipmin, yclipmax, zclipmin, zclipmax에 설정된다. 이 경계들은 임의의 값으로 설정될수 있지만 그것들은 보통 보임창의 경계들로 설정된다. 자르기루틴을 *xy*면 또는 보기체적의 앞뒤면(또는 정의된 자르기경계)에 대해 on/off절환하기 위하여 파라메터 clipxy, clipback, clipfront에 값 *clip* 또는 *noclip*를 할당한다.

자르기루틴을 무시하는것이 편리한 때가 있다. 장면의 초기구성시 물체들의 시험배치를 빨리 현시할수 있도록 자르기하지 않을수 있다. 또한 모든 물체들이 자르기면들의 내부에 있다는것을 알면 하나 또는 그이상의 자르기면들을 없앨수 있다.

보기표가 설정되면 워크스테이션에서 개개의 보기표현을 함수

```
setViewIndex (viewIndex)
```

에 의하여 선택한다. 보기첨수번호는 능동워크스테이션에서 후에 지적되는 출력기초요소들에 적용될 보기변환파라메터들의 모임을 식별한다.

마지막으로 워크스테이션에서의 현시를 위한 부분투영창문을 선택하는데 워크스테이션변환함수들을 리용할수 있다. 이 함수들은 창문과 보임창구역이 3차원구역이라는것을 제외하고 2차원보기에서 론의한것들과 류사하다. 창문함수는 단위바른6면체의 구역을 선택하며 보임창함수는 출력장치에 대한 현시구역을 선택한다. 창문의 경계들은 정규투영자리표로

```
setWorkstationWindow3 (ws, xwsWindmin, xwsWindmax,
    ywsWindmin, ywsWindmax, zwsWindmin, zwsWindmax)
```

에 의하여 설정되며 보임창의 경계들은 장치자리표로

```
setWorkstationViewport3 (ws, xwsVPortmin, xwsVPortmax,
    ywsVPortmin, ywsVPortmax, zwsVPortmin, zwsVPortmax)
```

에 의하여 설정된다. 그림 12-46은 PHIGS Toolkit쏘프트웨어를 리용하는 PHIGS보기경로에서 보기파라메터들의 대화식선택실례를 보여 준다.이 쏘프트웨어는 보기편집기, 창문, 차림표, 기타 대면도구들을 가지는 PHIGS대면부를 제공하기 위하여 만체스터종합대학에서 개발하였다.

일부 응용들에서는 카메라의 방향이 다른 보임상들을 결합하여 현시하는데 합성방법들을 리용한다. 그림 12-47은 가상현실환경을 위하여 만든 광각원근보임상을 보여 준다. 넓은 시야는 동일한 보기위치에서 보기방향을 약간씩 옮겨 장면의 7개 보임상을 만들어 얻는다.

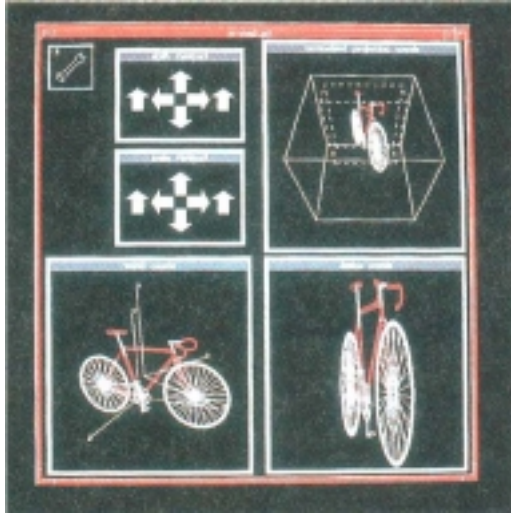



그림 12-46. 보기경로에서 파라메터들을 대화식으로 조종하기 위하여 만체스터종합대학에서 개발한 PHIGS Toolkit의 리용

그림 12-47. 가상현실현시를 위하여 보기방향이 조금씩 다른 7개의 부분보임상들로 만든 광각보임상

요약

3차원장면에 대한 보기절차들은 2차원보기에서 리용되는 일반적인 방법을 따른다. 즉 먼저 모형화자리표에 의한 물체정의로부터 세계자리표장면을 만든다. 다음에 보기자리표계를 설치하고 물체표현을 세계자리표로부터 보기자리표에로 넘긴다. 마지막에 보기자리표는 장치자리표로 변환된다.

그러나 2차원보기와 달리 3차원보기는 물체표현을 장치자리표에로의 변환전에 보기면에로 변환하는 투영루틴들을 요구한다. 또한 3차원보기조작들은 더 많은 공간파라메터들을 포함한다. 3차원의 보기파라메터들을 서술하는데 카메라의 위치와 방향 같은것들을 리용할수 있다. 보기자리표계는 보기참조점, 보기면의 법선벡토르, 보기의 웃방향벡토르에 의하여 설정된다. 보기면의 위치는 보기축을 따라 설정되며 물체표현은 이 면에 투영된다. 물체표현을 보기면에로 넘기는데 원근투영 또는 평행투영방법들을 리용할수 있다.

평행투영에는 정투영과 비탈투영이 있으며 투영벡토르에 의하여 지적할수 있다. 물체의 하나이상의 면들을 현시하는 정투영을 축측투영이라고 한다. 물체의 등측투영은 매개 주축을 같은 량의 원근으로 그리는 축측투영에 의하여 얻어 진다. 일반적으로 리용되는 비탈투영은 캐빌리어투영과 캐비니트투영이다. 물체의 원근투영은 투영참조점에서 만나는 투영선들에 의하여 얻어 진다.

3차원장면안의 물체들은 보기체적으로 자른다. 보기체적의 웃면, 밑면, 옆면들은 투영선에 평행이며 보기면의 창문변들을 지나는 평면들에 의하여 만들어 진다. 닫긴 보기체적을 만들기 위하여 앞면과 뒤면을 리용한다. 평행투영에서 보기체적은 평행6면체이며 원근투영에서 보기체적은 각추대이다. 3차원보기에서 물체들은 보기체적의 경계면들에 대하여 물체의 자리표들을 검사하는것에 의해 잘라진다. 자르기는 일반적으로 도형처리프로그람들에서 모든 보기와 기타 변환들이 수행된후 동차자리표에 의하여 수행된다. 다음에 동차자리표는 3차원직각자리표로 변환된다.

참고문헌

PHIGS및 PHIGS+ 에서 3차원의 보기조작과 자르기조작의 추가적인 정보는 Howard 등(1991), Gaskins(1992), Blake(1993)에서 보시오. 3차원의 자르기알고리듬과 보기알고리듬은 Blinn과 Newell(1978), Cyrus와 Beck(1978), Riesenfeld(1981), Liang과 Barsky(1984), Arvo(1991), Blinn(1993)에서 보시오.

련습문제

12-1. 식 12-2~12-4를 리용하여 함수 evaluateViewOrientationMatrix3을 수행하는 절차를 쓰시오.

12-2. 함수 setViewRepresentation3과 setViewIndex를 수행하는 루틴들을 쓰시오.

12-3. 투영벡토르가 지적되였을 때 평행투영을 리용하여 다면체의 정점들을 투영자리표로 변환하는 절차를 쓰시오.

12-4. 지적된 회전을 적용하고 다면체의 여러가지 평행투영을 얻는 절차를 쓰시오.

12-5. 물체의 한점원근투영을 수행하는 절차를 쓰시오.

12-6. 물체의 두점원근투영을 수행하는 절차를 쓰시오.

12-7. 물체의 세점원근투영을 수행하는 루틴을 개발하시오.

12-8. 원근투영의 각추대를 정규직6면체로 변환하는 루틴을 쓰시오.

12-9. 3차원평면을 정규직6면체로 자르기 위하여 싸더랜드-호쥐맨의 다각형자르기알고리듬을 확장하시오.

12-10. 장면안의 물체를 정의된 각추대로 자르는 알고리듬을 창안하시오. 이 알고리듬에 필요한 조작들을 정규직6면체로 자르는 알고리듬에 필요한 조작들과 비교하시오.

12-11. 3차원의 선들을 지적된 정규직6면체로 자르기 위하여 2차원의 리앙-바스키선자르기알고리듬을 수정하시오.

12-12. 주어 진 다면체를 지적된 정규직6면체로 자르기 위하여 2차원의 리앙-바스키선자르기알고리듬을 수정하시오.

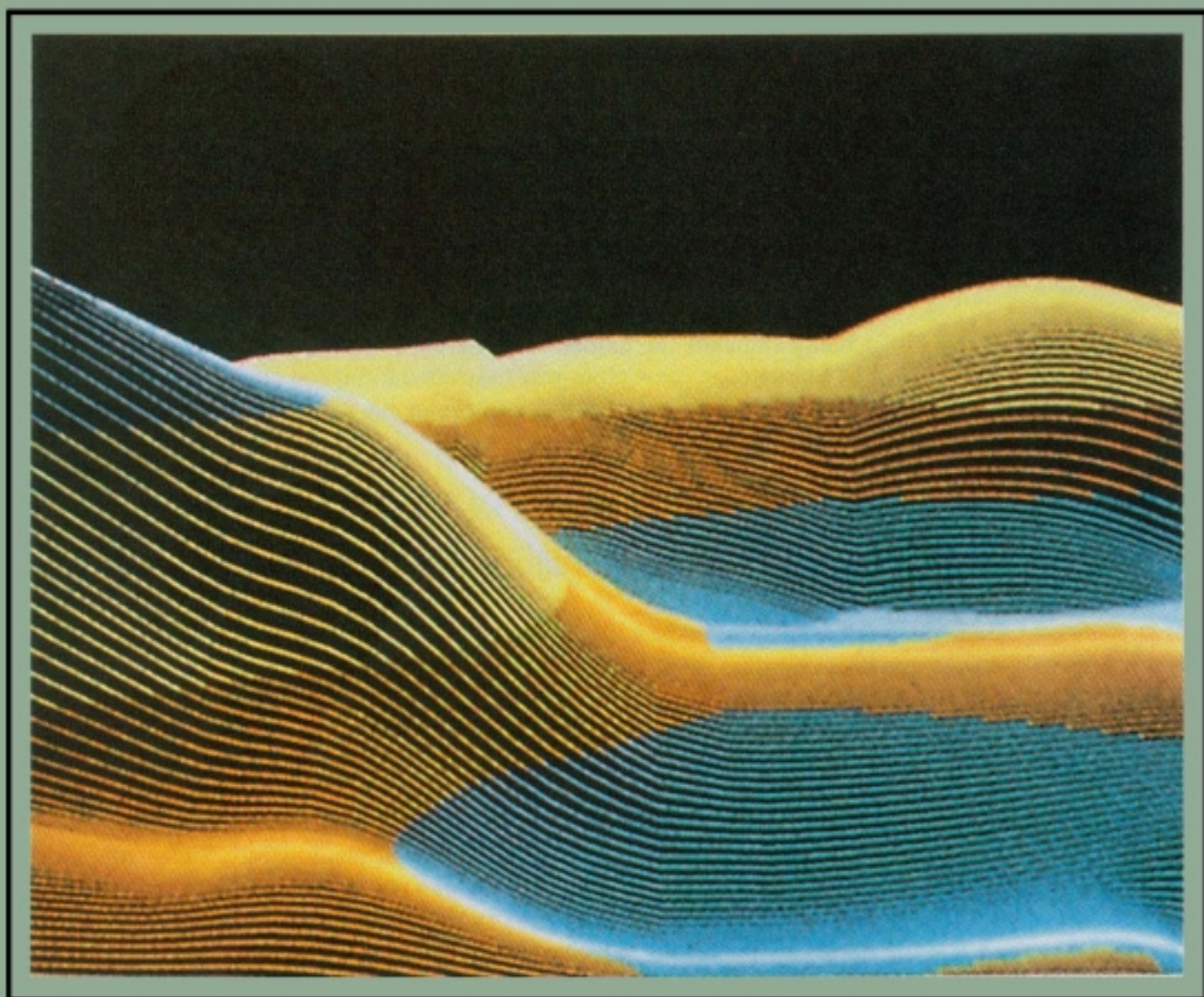
12-13. 다면체를 평행6면체로 자르기 위한 알고리듬을 작성하시오.

12-14. 동차자리표에 의하여 자르기하는 루틴을 쓰시오.

12-15. 임의의 자르기절차와 정투영을 리용하여 세계자리표로부터 장치자리표에로의 완전한 보기변환을 수행하는 프로그람을 쓰시오.

12-16. 임의의 자르기절차와 임의로 지적된 평행투영벡토르를 리용하여 세계자리표로부터 장치자리표에로 완전한 보기변환을 수행하는 프로그람을 쓰시오.

12-17. 원근투영에 대하여 보기경로안의 모든 걸음들을 수행하는 프로그람을 쓰시오.

13장. 보이는 면의 검출방법

도형처리를 리용하여 장면을 현실감 있게 현시하는데서 중요한것은 선택된 보기위치에서 장면의 보이는 부분들을 식별하는것이다. 이 문제를 풀수 있는 여러가지 방법들이 있는데 서로 다른 류형의 응용들에서 보이는 물체들을 효률적으로 식별하기 위한 수값알고리듬들이 창안되였다. 일부 방법들은 많은 기억기를 요구하고 다른 방법들은 처리시간이 길며 또 다른 방법들은 특별한 류형의 물체들에만 적용된다. 개개의 응용들에서는 장면의 복잡성, 현시될 물체의 류형, 사용가능한 설비, 정적으로 현시하는가 또는 동화식으로 현시하는가 하는 인자들에 따라 방법들을 결정한다. 이 여러가지 알고리듬들을 보이는 면의 검출방법(visible-surface detection method)이라고 한다. 때때로 이 방법들을 보이지 않는 면소거방법(hidden-surface elimination method)이라고도 말하지만 보이는 면의 식별과 보이지 않는 면의 소거사이에는 미묘한 차이가 있을수 있다. 실례로 선그물구조현시에서 보이지 않는 면들을 실제로 없애지 않고 오히려 그것들의 경계선을 파선으로 현시하거나 또는 그것들의 형태정보를 보유하는 다른 방법들로 현시하려 할수 있다. 이 장에서는 3차원장면에서 보이는 면의 검출에 제일 일반적으로 리용하는 몇가지 방법들을 연구한다.

1절. 보이는 면검출알고리듬의 분류

보이는 면을 검출하는 알고리듬들은 물체의 정의를 직접 취급하는가 또는 그의 투영된 상을 취급하는가에 따라 크게 두가지로 분류한다. 이 두가지 방법을 각각 물체공간방법과 화상공간방법이라고 한다. 물체공간방법은 전체적으로 보인다고 표식하여야 할 면들을 결정하기 위하여 장면안에서 정의된 물체들과 그것들의 부분들을 서로서로 비교한다. 화상공간알고리듬에서 보임성은 투영면의 매 화소위치에서의 점에 의하여 결정된다. 보이는 면을 검출하는 대부분의 알고리듬들은 화상공간방법을 리용하지만 일부 경우 보이는 면들을 찾아 내는데 물체공간방법이 효과적으로 리용될수 있다. 한편 선현시알고리듬들은 선그물구조현시에서 보이는 선들을 식별하는데 일반적으로 물체공간방법을 리용하지만 보이는 면을 검출하는 많은 화상공간알고리듬들도 보이는 선의 검출에 쉽게 적응될수 있다.

보이는 면을 검출하는 여러가지 알고리듬들이 취하는 기초적인 방법에서 중요한 차이가 있지만 대다수는 실행을 개선하기 위하여 정렬 및 밀착방법들을 리용한다. 정렬방법은 장면안의 개별적인 면들을 보기면으로부터의 거리에 따라 순서화하여 깊이비교를 쉽게 하는데 리용된다. 밀착방법은 장면의 규칙성의 우점을 취하는데 리용된다. 개별적인 주사선에서는 일정한 화소세기들의 구간(행정)을 취할수 있으며 주사선무늬는 한 주사선에서 다른 주사선으로 약간씩 변한다. 동화의 프레임들은 움직이는 물체의 근방에서만 변한다. 그리고 장면안의 물체와 면들사이에는 일정한 관계가 설정될수 있다.

2절. 뒤면검출

다면체의 뒤면들을 식별하는 빠르면서도 간단한 물체공간방법은 10장에서 설명한 《내부-외부》검사에 기초한다. 점 (x, y, z)는

$$Ax + By + Cz + D < 0 \tag{13-1}$$

이면 파라메터 A, B, C, D를 가지는 다각형면의 내부에 있다. 다각형면에 대한 조준선에서 보기위치가 다각형면의 내부점이면 그 다각형의 뒤면을 보게 된다(즉 현재의 보기위치로부터 그 다각형의 앞면을 볼수 없다.).

이 검사는 직각자리표성분 (A, B, C)를 가지는 다각형면의 법선벡토르 $\mathbf{N}$을 고찰하여 간단화할수

있다. 일반적으로 **V**가 그림 13-1에 보여 준바와 같이 눈(또는 카메라)위치로부터의 보기방향벡토르이면

$$\mathbf{V}\cdot\mathbf{N}>0 \qquad (13-2)$$

일 때 이 다각형면은 뒤면이다. 더우기 물체표현이 투영자리표에로 전환되고 보기방향이 z_v보기축에 평행이면 $\mathbf{V}=(0,\ 0,\ V_z)$

$$\mathbf{V}\cdot\mathbf{N}=V_z C$$

이다. 그러므로 법선벡토르 **N**의 z성분 C의 부호만을 고찰하면 된다.

보기방향이 부의 z_v축방향인 오른손보기자리표계(그림 13-2)에서 다각형면은 C<0이면 뒤면이다. 또한 법선의 z성분이 C=0인 면은 볼수 없다. 왜냐하면 보기방향이 그 다각형면을 스치기때문이다. 이리하여 일반적으로 다각형면의 법선벡토르가 z성분값

$$C\le 0 \qquad (13-3)$$

을 가지면 그 다각형면을 뒤면으로 표식할수 있다.

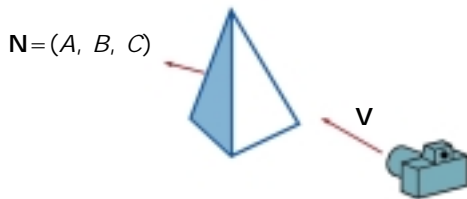



그림 13-1. 보기방향벡토르 **V**와 다면체의 뒤면의 법선벡토르 **N**

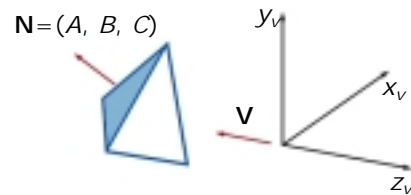



그림 13-2. 오른손보기자리표계에서 평면의 파라메터 C<0인 다각형면은 보기방향이 부의 z_v축방향일 때 뒤면으로 식별된다.

왼손보기자리표계를 쓰는 패키지들에서도 류사한 방법을 리용할수 있다. 이 패키지들에서 평면의 파라메터 A, B, C, D는 (오른손자리표계에서 리용하는 시계바늘반대방향대신에) 시계바늘방향으로 지적되는 다각형의 정점자리표들로부터 계산할수 있다. 그러면 부등식 13-1의 검사는 내부점들에 대하여 여전히 유효하다. 또한 뒤면은 보기위치로부터 멀어 져 감을 가리키는 법선벡토르를 가지며 보기방향이 정의 z_v축방향일 때 $C\ge 0$에 의하여 식별된다.

물체를 정의하는 면들에 대하여 파라메터 C를 검사하여 모든 뒤면들을 곧 식별할수 있다. 그림 13-2의 각추와 같은 단일한 볼록다면체에 대하여 이 검사는 물체의 모든 보이지 않는 면들을 식별한다. 왜냐하면 매개 면은 완전히 보이거나 또는 완전히 감추어 지기때문이다. 또한 장면이 겹치지 않는 볼록다면체들만을 포함하면 모든 보이지 않는 면들은 뒤면방법에 의하여 식별된다.

그림 13-3의 오목다면체와 같은 물체인 경우에는 다른 면들에 의해 전체적으로 또는 부분적으로 감추어 지는 추가적인 면들이 있는가를 결정하기 위하여 더 많은 검사를 하여야 한다. 그리고 일반적으로 장면에서 물체들은 조준선을 따라 겹칠수 있다. 그러면 감추어 지는 물체에서 다른 물체들에 의해 부분적으로 또는 완전히 감추어 지는 곳을 결정하여야 한다. 일반적으로 뒤면제거를 리용하면 장면에서 보임성검사를 진행하여야 할 다각형면들의 약 절반을 없앨수 있다.

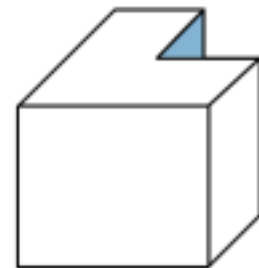



그림 13-3. 한 면이 다른 면에 의하여 부분적으로 감추어 지는 오목다면체

3절. 깊이완충기방법

보이는 면의 검출에 일반적으로 리용되는 화상공간방법은 깊이완충기방법(depth-buffer method)이다. 이 방법은 투영면의 매개 화소위치에서 면들의 깊이를 비교한다. 물체의 깊이는 보통 보기면으로부터 보기자리표계의 z축을 따라 측정되기때문에 이 방법을 z완충기방법(z-buffer method)이라고도 한다. 장면의 매면은 면을 가로질러서 한번에 한점씩 개별적으로 처리된다. 이 방법은 보통 다각형면들만을 가지는 장면에 적용된다. 왜냐하면 깊이값들을 대단히 빨리 계산할수 있으며 방법을 실현하기 쉽기때문이다. 그러나 이 방법은 곡면들에도 적용할수 있다.

투영자리표로 변환된 물체표현에 의하여 다각형면우의 매개 (x, y, z)위치는 보기면우의 정투영점 (x, y)에 대응한다. 따라서 보기면우의 매개 화소위치 (x, y)에 대하여 물체의 깊이는 z값을 비교하는 방법으로 비교할수 있다. 그림 13-4는 정투영선을 따라 x_v, y_v면으로 취한 보기면우의 위치 (x, y)로부터 서로 다른 거리에 있는 3개의 면들을 보여 준다. 면 S_1은 이 위치에 제일 가까우며 (x, y)에는 그 면의 세기값이 보관된다.

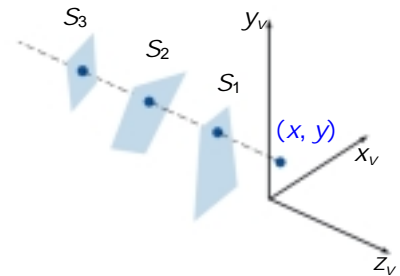



그림 13-4. 보기면우의 위치 (x, y)에서 면 S_1은 보기면으로부터 제일 작은 깊이를 가지며 따라서 그 위치에서 보인다.

깊이완충기알고리듬을 정규화된 자리표에서 실현할수 있으므로 z값은 뒤자르기면의 0부터 앞자르기면의 z_{max}까지의 범위에 있다. z_{max}의 값은 1(단위바른6면체에 대하여) 또는 체계에 기억될수 있는 제일 큰 값으로 설정될수 있다.

이 방법의 이름에 함축되여 있는바와 같이 두개의 완충기구역이 요구된다. 깊이완충기는 면이 처리될 때 매개 (x, y)위치에 대한 깊이값을 기억하는데 리용되며 재생완충기는 매개 위치에 대한 세기값을 기억한다. 초기에 깊이완충기의 모든 위치들은 0(최소깊이)으로 설정되며 재생완충기는 배경세기로 초기화된다. 다각형표에 기입된 매개 면은 한번에 한개 주사선씩 매개 (x, y)화소위치에서의 깊이(z값)를 계산하면서 처리된다. 계산된 깊이는 깊이완충기의 그 위치에서 이미 기억된 값과 비교된다. 계산된 깊이가 깊이완충기에 기억된 값보다 크면 새로운 깊이값이 기억되며 그 위치에서의 면의 세기가 결정되고 재생완충기의 동일한 xy위치에 넣어 진다.

깊이완충기알고리듬의 걸음들을 다음과 같이 요약한다.

1. 깊이완충기와 재생완충기를 완충기의 모든 위치 (x, y)에 대하여

$$\text{depth}(x, y)=0, \quad \text{refresh}(x, y)=I_{\text{backgnd}}$$

   으로 초기화한다.
2. 매개 다각형면우의 매 위치에 대하여 보임성을 결정하기 위하여 깊이값을 깊이완충기에서 이미 기억된 값과 비교한다.

   - 다각형우의 매개 (x, y)위치에 대하여 깊이 z를 계산한다.
   - $z>\text{depth}(x, y)$이면

$$\text{depth}(x,\ y) = z,\ \ \text{refresh}(x,\ y) = \text{I}_{\text{surf}}(x,\ y)$$

로 설정한다. 여기서 $I_{backgnd}$는 배경세기에 대한 값이고 $I_{surf}(x,\ y)$는 화소위치$(x,\ y)$에서 면에 대한 투영된 세기값이다. 모든 면들이 처리된후 깊이완충기는 보이는 면들에 대한 깊이값을 포함하고 재생완충기는 그 면들에 대한 대응하는 세기값을 포함한다.

위치 $(x,\ y)$에 대한 면의 깊이값은 매개 면에 대한 평면의 방정식으로부터 계산된다.

$$z = \frac{-Ax - By - D}{C} \qquad (13-4)$$

임의의 주사선에서(그림 13-5) 그 선우의 린접한 수평위치들은 1만큼 차이나며 린접한 주사선들에서 수직 y값은 1만큼 차이난다. 위치(x, y)의 깊이가 z로 결정되면 그 주사선우의 다음위치 $(x+1, y)$의 깊이 z'는 식 13-4로부터

$$z' = \frac{-A(x+1) - By - D}{C} \qquad (13-5)$$

또는

$$z' = z - \frac{A}{C} \qquad (13-6)$$

와 같이 얻어 진다. 비 $-A/C$는 매개 면에서 일정하며 주사선우의 련속된 깊이값들은 앞의 값으로부터 단일더하기를 하여 얻는다.

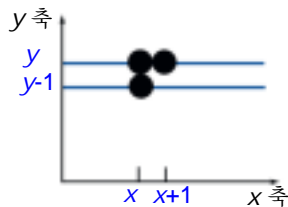



그림 13-5. 주사선우의 위치 (x, y)로부터 그 선우의 다음위치는 자리표$(x+1, y)$를 가지며 그아래 다음 주사선우의 위치는 자리표 $(x, y-1)$을 가진다.

계산은 매개 주사선에서 그 주사선과 사귀는 다각형의 왼쪽 변에서의 깊이를 계산하는 방법으로 시작한다(그림 13-6). 이때 그 주사선을 따라 매 련속된 위치들에서의 깊이값은 식 13-6에 의하여 계산된다.

그림 13-6. 다각형면을 가로지르는 주사선들

먼저 매개 다각형의 y자리표범위를 결정하고 그림 13-6에 보여 준바와 같이 면을 제일 꼭대기주사선으로부터 아래주사선에로 가면서 처리한다. 꼭대기정점에서 시작하여 다각형의 왼쪽 변을 따라 내려 가면서 x위치를 $x' = x-1/m$과 같이 재귀적으로 계산할수 있다. 여기서 m은 변의 경사도이다(그림 13-7). 변을 따라 내려 갈 때 깊이값은 재귀적으로

$$z' = z + \frac{A/m + B}{C} \qquad (13-7)$$

와 같이 얻어 진다.

수직변을 따라 처리하면 경사도는 무한대이며 재귀계산은

$$z' = z + \frac{B}{C}$$

로 줄어 든다.

또 다른 방법은 매개 주사선의 왼쪽 변우의 x값을 결정하기 위하여 중점방법 또는 브리센함형의 알고리듬을 리용하는것이다. 또한 이 방법은 매개 면의 투영점들에서의 깊이 및 세기값을 결정하는 방법으로 곡면에 적용할수 있다.

그림 13-7. 다각형의 왼쪽 변과 주사선들과의 사귐점위치들

다각형면에 대한 깊이완충기방법은 실현하기가 대단히 쉽고 장면안의 면들을 정렬시키지 않아도 된다. 그러나 그것은 재생완충기외에 두번째 완충기를 요구한다. 실례로 분해능이 1024×1024인 체계는 깊이완충기에 약 백만개의 깊이값을 기억하여야 하며 깊이값을 표현하는데 충분한 비트를 가져야 한다. 기억용량을 줄이는 한가지 방법은 더 작은 깊이완충기를 리용하여 한번에 장면의 일부분을 처리하는것이다. 매개 보기부분이 처리된후 완충기는 다음부분에 다시 리용된다.

4절. A완충기방법

깊이완충기방법의 확장은 A완충기방법이다(z는 깊이를 표현한다. 《z완충기》로부터 자모의 다른 끝에는 A가 있다.). A완충기방법은 REYES(《Renders Everything You Ever Saw》의 준말)라고 하는 면실감처리체계의 실현을 위하여 Lucasfilm이 개발한 경계허상이 제거된 구역평균축적완충기방법을 표현한다.

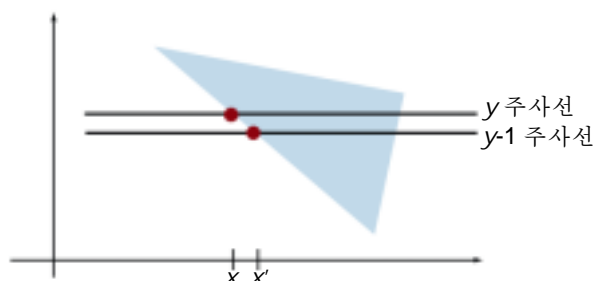



그림 13-8. 투명한 면을 통해 불투명한 면을 보기 위하여 화소위치에서 여러개 면들의 세기를 고려하여야 한다.

깊이완충기방법의 결함은 매개 화소위치에서 한개의 보이는 면만을 찾을수 있다는것이다. 다시말하여 불투명한 면들만을 취급하며 투명한 면들이 현시될 때에는(그림 13-8) 여러개의 면들에 대한 세기값들을 축적할수 없다. A완충기방법은 완충기의 매개 위치가 면들의 련결목록을 참조할수 있도록 깊이완충기를 확장한다. 이리하여 매개 화소위치에서 하나이상의 면세기들이 고찰될수 있으며 물체변두리의 경계허상이 제거될수 있다.

A완충기의 매개 위치는 두개의 마당을 가진다.

- 깊이마당 - 정 또는 부의 실수를 기억한다.
- 세기마당 - 면세기정보 또는 지시자값을 기억한다.

깊이마당이 정이면 그 위치에 기억된 수는 대응하는 화소구역에 겹치는 한개 면의 깊이이다. 그러면

세기마당은 그림 13-9 ㄱ에서 설명하는바와 같이 그 점에서 면색갈의 RGB성분들과 화소적용범위의 퍼센트를 기억한다.

깊이마당이 부이면 이것은 화소세기에 여러개의 면들이 관계된다는것을 지적한다. 그러면 세기마당은 그림 13-9 ㄴ에서와 같이 면자료의 련결목록에 대한 지시자를 기억한다. 련결목록안의 매면의 자료는

- RGB세기성분들
- 불투명도파라메터(투명도의 퍼센트)
- 깊이
- 구역적용범위의 퍼센트
- 면식별자
- 다른 면실감처리파라메터들
- 다음면에 대한 지시자

를 포함한다.

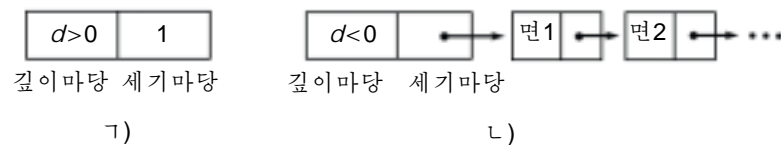



그림 13-9. A완충기의 화소위치의 구성(대응하는 화소구역에는 한개 면(ㄱ) 또는 여러개의 면(ㄴ)이 있다.)

A완충기는 깊이완충기알고리듬과 류사한 방법으로 만들수 있다. 개별적인 주사선을 따라 화소의 면겹침을 결정하도록 주사선들을 처리한다. 면들은 다각형그물로 부분분할되며 화소경계에서 잘라 진다. 불투명도인자와 면겹침의 퍼센트를 리용하여 매개 화소의 세기를 겹침면들에 대한 평균으로 계산할수 있다.

5절. 주사선방법

보이지 않는 면을 제거하는 이 화상공간방법은 다각형내부를 채우는 주사선알고리듬의 확장이다. 한면을 채우는것대신에 다중면을 취급한다. 매개 주사선이 주사될 때 그 주사선과 사귀는 모든 다각형면들은 어느것이 보이는가를 결정하기 위하여 검사된다. 매개 주사선을 가로질러서 어느것이 보기면에 제일 가까운가를 결정하기 위하여 매개 겹침면에 대해 깊이를 계산한다. 보이는 면이 결정될 때 그 위치에 대한 세기값이 재생완충기에 들어 간다.

10장에서 설명한바와 같이 여러가지 면들에 대하여 표가 설정된다고 하자. 그것들은 변표와 다각형표를 다같이 포함한다. 변표는 장면안의 매선에 대한 자리표끝점들, 매선의 거꿀경사도, 매선에 의하여 경계 지어지는 면을 식별하는 다각형표에로의 지시자를 포함한다. 다각형표는 매면에 대한 평면방정식의 결수들과 면에 대한 세기정보, 가능하다면 변표에로의 지시자까지도 포함한다. 주어 진 주사선과 교차하는 면조사를 쉽게 하기 위하여 변표의 정보로부터 변들의 능동목록을 설정할수 있다. 이 능동목록은 현재의 주사선과 교차하며 x가 증가하는 순서로 정렬된 변들만을 포함한다.

게다가 주사선우의 위치가 면의 내부인가 외부인가를 지적하기 위하여 매면에 대하여 on 또는 off로 설정되는 기발을 정의한다. 주사선들은 왼쪽에서 오른쪽으로 처리된다. 면의 제일 왼쪽 경계에서 면 기발은 on으로 되고 제일 오른쪽 경계에서 off로 된다.

그림 13-10은 주사선우의 화소위치들에 대하여 면의 보이는 부분을 찾아내기 위한 주사선방법을 설명한다. 주사선 1에 대한 능동목록은 변표로부터 변 *AB*, *BC*, *EH*, *FG*에 대한 정보를 포함한다. 이 주사선을 따라 변 *AB*와 *BC*사이의 위치들에 대하여 면 S_1에 대한 기발만이 on이다. 따라서 깊이계산은 필요 없으며 면 S_1에 대한 세기정보는 다각형표로부터 재생완충기에로 들어 간다. 류사하게 변 *EH*와 *FG*사이에서 면 S_2에 대한 기발만이 on이다. 주사선 1우의 다른 위치들은 면들을 가로지르지 않으므로 이 위치들에서의 세기값은 배경세기로 설정된다. 배경세기는 초기화루틴에서 완충기에 적재될수 있다.

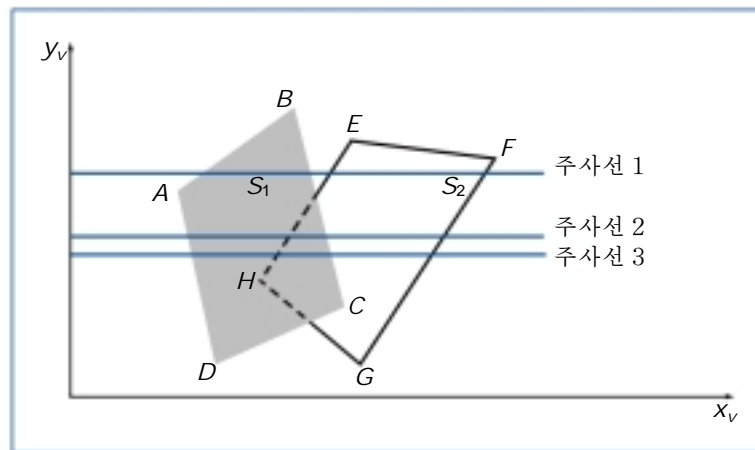



그림 13-10. 보기면에서 두 면 S_1와 S_2의 투영을 가로지르는 주사선들(파선은 숨은 면의 경계를 지적한다.)

그림 13-10에서 주사선 2와 3에 대한 능동변목록은 변 *AD*, *EH*, *BC*, *FG*를 포함한다. 주사선 2를 따라 변 *AD*로부터 변 *EH*까지 면 S_1에 대한 기발만이 on이다. 그러나 변 *EH*와 *BC*사이에서는 두 면에 대한 기발이 다같이 on이다. 이 구간에서 두 면에 대한 평면의 결수들을 리용하여 깊이계산을 하여야 한다. 이 실례에서 면 S_1의 깊이는 S_2의것보다 작다고 가정되며 따라서 면 S_1에 대한 세기가 경계 *BC*를 만날 때까지 재생완충기에 적재된다. 다음에 면 S_1에 대한 기발은 off로 되며 면 S_2에 대한 세기는 변 *FG*를 통과할 때까지 기억된다.

한 주사선으로부터 다음주사선에로 통과할 때 주사선을 따라 밀착성의 우점을 리용할수 있다. 그림 13-10에서 주사선 3은 주사선 2와 같은 능동변목록을 가진다. 선의 사귐점들에서 변화가 일어나지 않았기때문에 변 *EH*와 *BC*사이에서 깊이계산은 다시 하지 않아도 된다. 두 면은 주사선 2에서 결정된것과 같은 방향에 있어야 하며 따라서 면 S_1에 대한 세기는 다시 계산하지 않고 넣을수 있다.

임의의 개수의 겹침다각형면들을 이 주사선방법에 의하여 처리할수 있다. 면에 대한 기발은 위치가 내부 또는 외부인가를 지적하도록 설정되며 깊이계산은 면들이 겹칠 때 수행된다. 이 밀착방법을 리용할 때 매개 주사선에서 어느 면의 부분이 보이는가 하는 자리길을 보존하는데 주의하여야 한다. 이것은 면들이 자르지 않거나 그렇지 않으면 서로 반복적으로 겹치지 않을 때(그림 13-11)에만 작업한다. 임의의 종류의 주기적인 겹침이 장면안에 있으면 겹침을 없애기 위하여 면을 나눌수 있다. 이 그림에서 파선은 두개의 서로 다른 면을 형성하기 위하여 면이 부분분할될수 있는 곳을 지적하므로 주기적인 반복이 없어 진다.

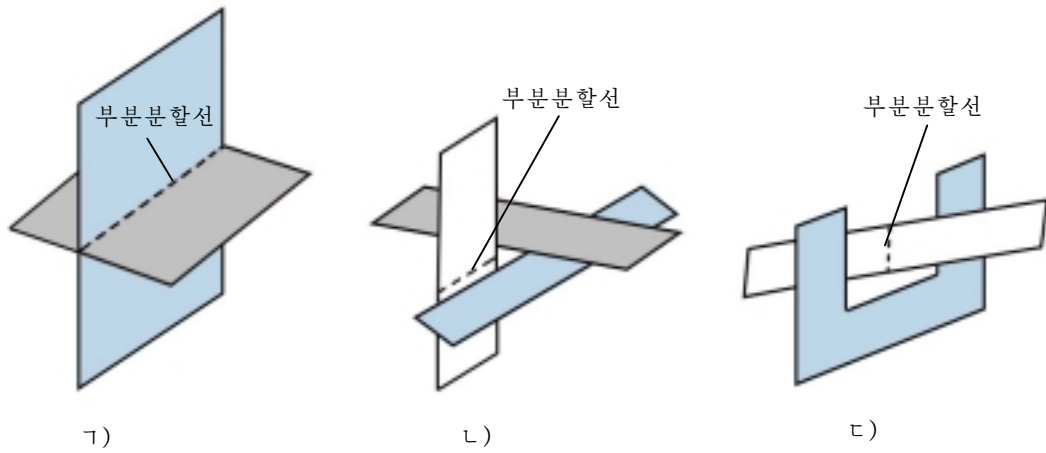



그림 13-11. 가로지르는 면 및 엇바꾸어 겹치는 주기적인 겹침면

6절. 깊이정렬방법

화상공간 및 물체공간조작들을 리용하여 깊이정렬방법(depth-sorting method)은 다음의 기초적인 기능들을 수행한다.

1. 면들은 깊이가 감소하는 순서로 정렬된다.
2. 면들은 제일 큰 깊이의 면으로부터 시작하여 순서대로 주사전환된다.

정렬조작은 화상 및 물체공간에서 다같이 실행되며 다각형면의 주사전환은 화상공간에서 실행된다.

보이지 않는 면소거문제를 해결하는 이 방법은 자주 화가의 알고리듬이라고 한다. 유화를 그릴 때 화가는 먼저 배경색을 그린다. 다음에 제일 먼 물체를 그리며 그후 더 가까운 물체들을 그린다. 마지막걸음에서 배경과 화포우에 그려 진 다른 물체들우의 화포에 전경물체를 그린다. 매개 색칠층은 앞의 층을 덮는다. 류사한 수법을 리용하여 먼저 면들을 보기면으로부터의 거리에 따라 정렬시킨다. 제일 먼 면에 대한 세기값이 재생완충기에 넣어 진다. 매개 뒤따르는 면을 차례로(깊이가 감소하는 순서로) 취하여 프레임완충기에서 이미 처리된 면의 세기우에 그 면의 세기를 《색칠한다》.

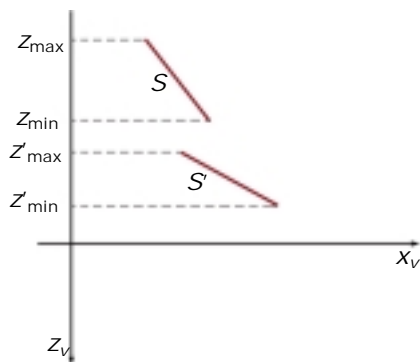



그림 13-12. 깊이겹침이 없는 두 면

프레임완충기에서 다각형면들을 깊이에 따라 색칠하는것은 여러 걸음으로 수행된다. z방향을 따라 본다고 하면 면들은 매개 면의 제일 큰 z값에 따라 첫번째 경로에서 순서화된다. 제일 큰 깊이를 가지는 면 S는 다음에 깊이에서의 어떤 겹침이 있지 않는가를 결정하기 위하여 목록안의 다른 면들과 비교된다. 깊이겹침이 없으면 S는 주사전환된다. 그림 13-12는 깊이겹침은 없지만 xy면에서 겹치는 두 면을 보여 준다. 다음 이 처리는 목록안의 다음 면에 대하여 반복된다. 겹침이 일어나지 않는 한 매개 면은 모두 주사전환될 때까지 깊이순서로 처리된다. 목록안의 어

떤 점에서 깊이겹침이 검출되면 어떤 면들이 재순서화되여야 하는가를 결정하기 위하여 추가적인 비교를 하여야 한다.

S와 겹치는 매개 면에 대하여 다음의 검사를 한다. 이 검사들중 어느 하나가 참이면 그 면에 대한 재순서화는 필요 없다. 검사는 어려워 지는 순서로 기입된다.

1. xy평면에서 두 면의 테두리직4각형들은 겹치지 않는다.
2. 면 S는 보기위치에 대하여 겹침면뒤에 놓인다.
3. 겹침면은 보기위치에 대하여 S의 앞에 놓인다.
4. 보기면에로의 두 면의 투영들은 겹치지 않는다.

이 검사는 기입된 순서로 진행하며 검사들중 하나가 참이라는것을 발견하면 다음겹침면에로 간다. 모든 겹침면들이 이 검사들중 적어도 하나를 통과하면 S면은 제일 뒤에 놓인다. 그러면 재순서화할 필요가 없으며 S는 주사변환된다.

검사 1은 두 부분으로 수행된다. 먼저 x방향에서의 겹침을 검사하고 다음에 y방향에서의 겹침을 검사한다. 이 두 방향중 어느 하나가 겹침을 보여 주지 않으면 두 평면은 서로서로 덮을수 없다. z방향에서 겹치지만 x방향에서 겹치지 않는 두 면의 실례를 그림 13-13에 보여 준다.

검사 2와 3은 《내부-외부》다각형검사에 의하여 수행할수 있다. 즉 S의 모든 정점들에 대한 자리표를 겹침면에 대한 평면의 방정식에 대입하고 결과부호를 검사한다. 면의 바깥이 보기위치쪽이 되도록 평면의 방정식이 설정될 때 S는 S의 모든 정점들이 S내부에 있으면(그림 13-14) S'의 뒤에 있다. 류사하게 S'의 모든 정점들이 S의 밖이면 S'는 완전히 S의 앞에 있다. 그림 13-15는 S의 완전히 앞에 있는 겹침면 S'를 보여 주는데 면 S는 완전히 S'내부에 있지 않다(검사 2는 참이 아니다.).

검사 1~3이 모두 실패하면 xy평면에서의 선의 방정식을 리용하여 두 면의 테두리변들사이의 사귐점을 검사하는 방법으로 검사 4를 진행한다. 그림 13-16에서 보여 주는바와 같이 두 면은 그의 자리표범위들이 x, y, z방향에서 겹치더라도 사귀거나 그렇지 않을수 있다.

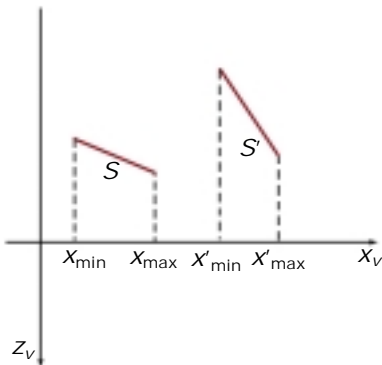



그림 13-13. 깊이겹침은 있지만 x방향에서의 겹침이 없는 두 면

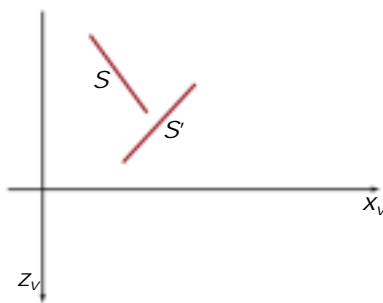



그림 13-14. 면 S는 완전히 겹침면 S'의 뒤(내부)에 있다.

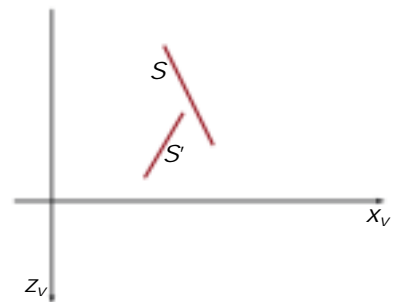



그림 13-15. 겹침면 S'는 면 S의 앞(외부)에 있지만 S는 S'의 완전히 뒤에 있지 않다.

4개의 모든 검사가 특별한 겹침면 S'에 의하여 실패하면 정렬목록에서 면 S와 S'를 교체한다. 이 절차에 의하여 재순서화된 두 면의 실례를 그림 13-17에서 보여 주었다. 이때 보기면으로부터 제일 먼 면을 정확히 찾았는가를 아직 모른다. 그림 13-18에서는 처음에 S와 S''를 교체하는 과정을 보여

주었다. 그러나 S''가 S'의 부분을 덮기때문에 세개 면의 정확한 깊이순서를 얻기 위하여 S''와 S'를 교체하여야 한다. 따라서 목록에서 재순서화된 매면의 검사처리를 반복하여야 한다.

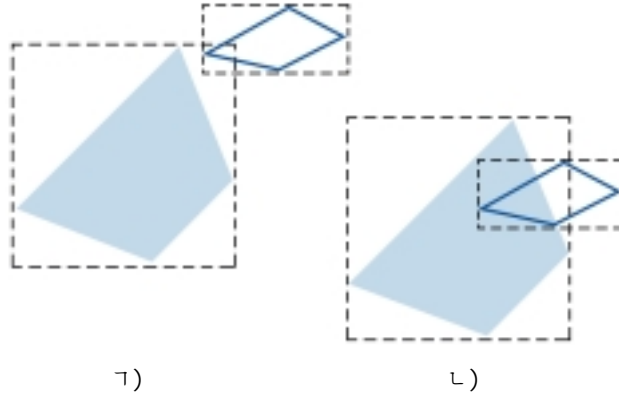



그림 13-16. xy평면에서 겹치는 테두리 직4각형을 가지는 두 면

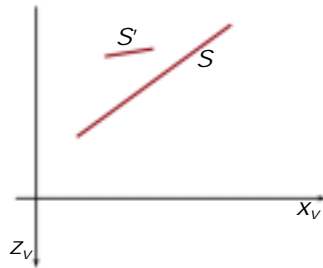



그림 13-17. 면 S는 더 큰 깊이를 가지지만 면 S'를 덮어 감춘다.

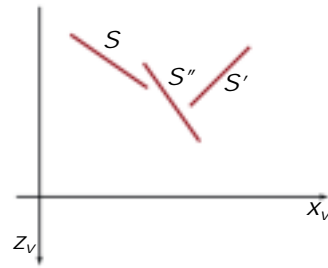



그림 13-18. 분류된 면 목록에 S, S', S''의 순서로 들어 있는 세개의 면은 S', S'', S로 재순서화되여야 한다.

그림 13-11에서와 같이 둘 또는 그이상의 면들이 서로서로 번갈아 덮어 감추면 방금 말한 알고리듬은 무한순환에 들어 갈수 있다. 이런 경우에 알고리듬은 겹침면들의 위치를 계속 교체한다. 이런 순환을 피하기 위하여 재순서화된 결과 더 먼 깊이위치에 있는 임의의 면을 다시 움직일수 없도록 기발표식할수 있다. 면을 두번 절환하는 시도가 이루어 지면 주기적인 겹침을 없애기 위하여 그것을 두 부분으로 나눈다. 본래의 면은 두개의 새면으로 교체되며 앞에서와 같이 처리를 계속한다.

7절. BSP나무방법

BSP나무방법은 화가의 그림그리기수법과 같이 뒤에서부터 앞으로 나오면서 면들을 화면에 색칠해 넣어 물체의 보임성을 결정하는 효률적인 방법이다. BSP나무방법은 보기참조점이 변해도 장면안의 물체들이 고정된 위치에 있을 때 대단히 편리하다.

BSP나무를 보임성검사에 적용할 때 공간부분분할의 매 걸음에서 보기방향에 대하여 분할면의 내부 및 외부에 있는 면들을 식별한다. 그림 13-19는 이 알고리듬의 기초적인 개념을 설명한다. 평면 P_1에 의하여 먼저 공간을 물체들의 두 모임으로 갈라 놓는다. 물체들의 한 모임은 보기방향에 대하여 평면 P_1의 뒤에 있으며 다른 모임은 P_1의 앞에 있다. 한 물체가 평면 P_1와 사귀기때문에 그 물체를 두개의 개별물체로 나누고 A 및 B로 표식한다. 물체 A와 C는 P_1의 앞에 있고 물체 B와 D는 P_1의 뒤에 있다. 다음에 공간을 다시 평면 P_2에 대하여 가르고 그림13-19 ㄴ에 보여 준 2분나무표현을 만든다. 이 나무에서 물체들은 말단마디로 표현되며 앞의 물체들은 왼쪽 가지로, 뒤의 물체들은 오른쪽 가지로 표현된다.

다각형면으로 표현된 물체들에 대하여 분할면을 다각형면과 일치하도록 선택한다. 다각형의 방정식은 내부 및 외부다각형들을 식별하는데 리용되며 나무는 매 다각형면에 대하여 하나씩의 분할면에 의해 만들어 진다. 분할면과 사귀는 임의의 다각형은 두 부분으로 분할된다. BSP나무가 완성될 때 현시를 위한 면들을 뒤에서부터 앞으로의 순서로 선택하여 나무를 처리하며 전경물체는 배경물체우에 색칠된다. 일부 체계들에서는 하드웨어적으로 실현하여 나무의 생성 및 처리를 빨리하고 있다.

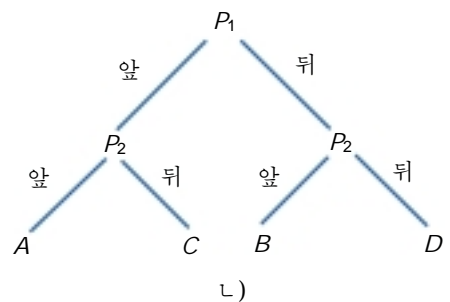
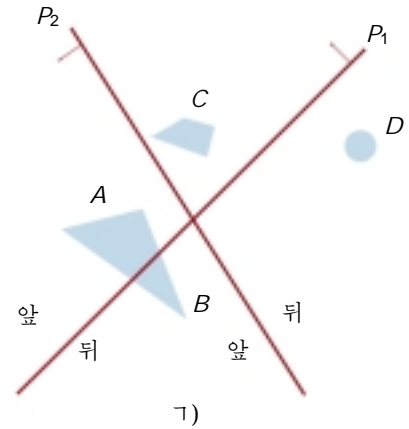



그림 13-19. 공간구역 ㄱ는 ㄴ의 BSP 나무표현을 만들기 위하여 두개의 평면 P_1와 P_2에 대해 갈라 진다.

8절. 구역부분분할방법

보이지 않는 면을 제거하는 이 방법은 본질적으로는 화상공간방법이지만 면들을 깊이에 따라 순서화하는데 물체공간조작들을 리용할수 있다. 구역부분분할방법(area-subdivision method)은 장면안의 개개의 부분면들을 표현하는 보기구역들을 찾아내여 구역들을 밀착시키는 우점을 가진다. 이 방법에서는 전체 보기구역을 소구역들로 분할하되 매 소구역에 보이는 부분면이 한개 있거나 또는 전혀 없을 때까지 점점 더 작은 직4각형들로 련속적으로 분할한다.

그림 13-20. 바른4각형구역을 매 걸음에서 같은 크기로 4등분한다.

이 방법을 실현하자면 구역이 단일면의 부분이라고 빨리 식별하거나 또는 구역이 분석하는데 너무 복잡하다고 쉽게 판정할수 있는 검사를 설정하여야 한다. 전체 보임상에서 시작하여 전체 구역을 더 작은 직4각형들로 부분분할하여야 하는가를 결정하는 검사를 적용한다. 검사에서 보임상이 아직 복잡하다고 판정되면 부분분할한다. 다음에 매개 작은 구역들에 검사를 적용하며 단일면의 보임성이 아직 복잡하다고 지적하면 다시 부분분할한다. 이 처리는 부분분할이 단일면에 속한다고 쉽게 분석될 때까지 또는 단일화소의 크기로 될 때까지 계속한다. 이렇게 하는 쉬운 방법은 그림 13-20에 보여 준바와 같이 구역을 매 걸음에서 4등분하는것이다. 이 방법은 4분나무

를 만드는 방법과 류사하다. 분해능이 1024×1024인 보기구역은 부분구역이 점으로 줄여 질 때까지 10번 부분분할된다.

지적된 구역안에서 단일면의 보임성을 결정하는 검사는 면을 구역의 경계와 비교하여 진행한다. 면과 지적된 구역경계사이에는 4가지 관계가 있을수 있다. 이 상대적인 면특성들을 다음의 방식(그림 13-21)으로 표현할수 있다.

둘러 싸는 면 - 구역을 완전히 둘러 싸는 면
겹침면 - 부분적으로 구역의 내부 및 외부에 있는 면
내부면 - 완전히 구역내부에 있는 면
외부면 - 완전히 구역외부에 있는 면

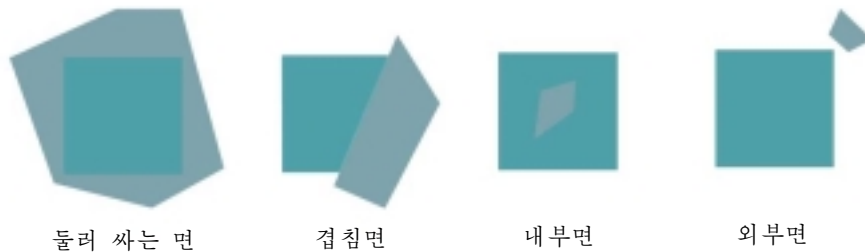



그림 13-21. 다각형면과 직4각형구역사이의 관계

구역안에서 면의 보임성을 결정하는 검사는 이 4분류에 의하여 진행할수 있다. 다음의 한개 조건이 참이면 지적된 구역을 앞으로 부분분할할 필요가 없다.

1. 모든 면들이 구역에 대하여 외부면이다.
2. 구역안에 한개의 내부면, 겹침면 또는 둘러 싸는 면만이 있다.
3. 둘러 싸는 면이 구역경계안의 다른 모든 면들을 덮어 감춘다.

검사 1은 모든 면들의 테두리직4각형을 구역경계에 대하여 검사하는 방법으로 실행할수 있다. 검사 2는 또한 내부면을 식별하기 위하여 *xy*평면에서의 테두리직4각형을 리용할수 있다. 다른 형의 면들에서는 테두리직4각형이 초기검사로 리용될수 있다. 만일 테두리직4각형이 어떤 방법으로 구역을 가로지르면 면이 둘러싸는가, 겹치는가 또는 외부인가를 결정하는데 추가적인 검사를 리용한다. 만일 내부, 겹침, 또는 둘러 싸는 면이 식별되면 그의 화소세기는 프레임완충기안의 적당한 구역에 이동된다.

검사 3을 수행하는 한가지 방법은 면들을 보기면으로부터의 최소깊이에 따라 순서화하는것이다. 매개 둘러 싸는 면에 대하여 고찰구역안에서의 최대깊이를 계산한다. 이 둘러 싸는 면들중 하나의 최대깊이가 구역안의 다른 모든 면들의 최소깊이보다 보기면에 더 가까우면 검사 3이 만족된다. 이 방법에 대한 실례를 그림 13-22에서 보여 주었다.

검사 3을 수행하는 깊이에 따르는 정렬을 요구하지 않는 다른 방법은 모든 둘러 싸는, 겹치는 내부면들에 대하여 구역의 네 정점에서의 깊이값을 계산하는데 평면의 방정식을 리용하는것이다. 둘러 싸는 면들중 하나에 대하여 계산된 깊이가 다른 모든 면들에 대하여 계산된 깊이보다 더 작으면 검사 3은 참이다. 그러면 구역은 둘러 싸는 면의 세기값으로 채워 질수 있다.

일부 경우 검사 3을 실현하는 두가지 방법은 다른 모든 면들을 덮어 감추는 둘러 싸는 면을 정확히 식별하는데 다같이 실패할것이다. 구역을 덮는 단일면을 식별하기 위하여 추가적인 검사를 수행할수 있지만 복잡한 검사를 계속하는것보다 구역을 부분분할하는것이 더 빠르다. 구역에 대한 외부 및 둘러 싸는 면이 식별되면 그것들은 구역의 모든 부분분할들에 대하여 외부 및 둘러 싸는 면으로 남을것이다. 게다가 부분분할처리를 계속할 때 일부 내부 및 겹침면들이 없어 질수 있으므로 구역들은 분석하기가 더 쉽다. 극단적인 경우에 화소크기의 부분분할이 만들어 지면 그 점에서 매개 관련면의 깊이를 간단히 계산하고 제일 가까운 면의 세기를 프레임완충기에 이동시킨다.

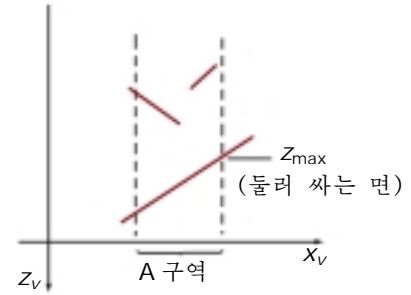



그림 13-22. 지적된 구역안에서 최대깊이 z_{max}를 가지는 둘러 싸는 면은 z_{max}이상의 최소깊이를 가지는 모든 면들을 덮어 감춘다.

기초적인 부분분할처리의 변종으로서 구역들을 절반으로 나누는 대신에 면경계를 따라 부분분할할수 있다. 면들이 최소깊이에 따라 정렬되였으면 제일 작은 깊이값을 가지는 면을 주어 진 구역을 부분분할하는데 리용할수 있다. 그림 13-23에서는 구역부분분할의 이 방법을 보여 주었다. 면 S의 경계의 투영은 본래의 구역을 부분분할 A_1 및 A_2로 가르는데 리용된다. 그러면 면 S는 A_1에 대하여 둘러 싸는 면이며 보임성검사 2와 3은 앞으로의 부분분할이 필요한가를 결정하는데 적용될수 있다. 일반적으로 이 방법을 리용하면 부분분할들이 더 적게 요구되지만 구역을 부분분할하고 부분분할의 경계에 대한 면들의 관계를 분석하는데 더 많은 처리가 필요된다.

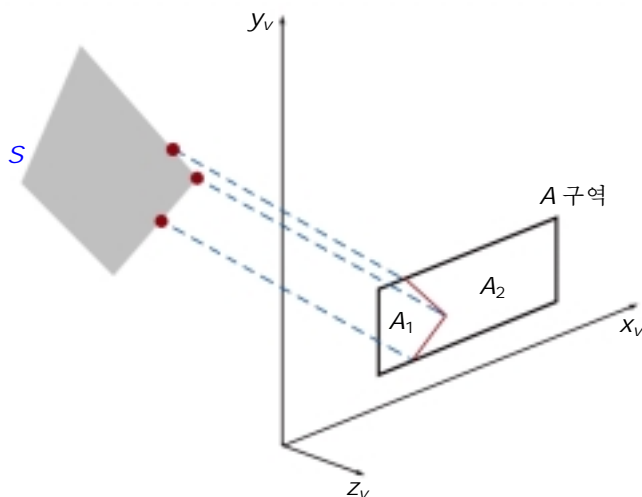



그림 13-23. 구역 A는 보기면에서의 면 S의 경계를 리용하여 A_1와 A_2로 부분분할된다.

9절. 8분나무방법

보기체적에 대하여 8분나무표현을 리용할 때 8분나무의 마디들을 앞뒤순서로 보기면에 투영하여 보이지 않는 면들을 소거한다. 그림 13-24에서 공간구역의 앞쪽(관찰자쪽)은 8분공간 0, 1, 2, 3에 의하여 형성된다. 이 8분공간들에서 앞쪽에 있는 면들은 관찰자에게 보인다. 앞쪽 8분공간들에서 뒤쪽에 있거나 또는 뒤쪽 8분공간들(4, 5, 6, 7)에 있는 임의의 면들은 앞쪽에 있는 면들에 의하여 감추어 질수 있다.

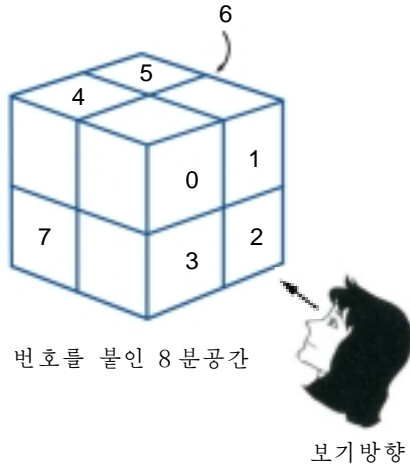



그림 13-24. 보기방향이 보여 준바와 같을 때 8분공간 0, 1, 2, 3에 있는 물체들은 뒤쪽 8분공간(4,5,6,7)에 있는 물체들을 덮어 감춘다.

그림 13-24에 주어 진 보기방향에 대하여 8분나무마디들의 자료요소들을 0, 1, 2, 3, 4, 5, 6, 7의 순서로 처리하여 뒤면들을 소거한다. 이것은 8분나무를 깊이우선순회하는것이며 그리하여 전체 공간구역에 대하여 8분공간 0, 1, 2, 3을 표현하는 마디들을 8분공간 4, 5, 6, 7을 표현하는 마디들보다 먼저 방문한다. 류사하게 8분공간 0에서 4개의 앞쪽 부분8분공간들에 대한 마디들을 4개의 뒤쪽 부분8분공간들에 대한 마디들보다 먼저 방문한다. 부분분할된 매개 8분공간에 대하여 이 순서로 8분나무를 계속 순회한다.

8분나무마디에서 색값을 만날 때 이 마디에 대응하는 프레임완충기안의 화소구역에는 이 화소구역에 앞서 기억된 값이 없을 때에만 그 색값을 할당한다. 이 방법에 의하여 앞쪽의 색갈들만을 프레임완충기에 적재한다. 화소구역이 무효이면 아무것도 적재하지 않는다. 완전히 덮어 감추어 진다고 발견되는 임의의 마디는 앞으로의 처리에서 없애며 따라서 그의 부분나무들은 호출되지 않는다.

8분나무로 표현되는 물체들의 서로 다른 보임상은 선택된 보임상에 따라 물체의 방향을 다시 정하는 변환을 8분나무표현에 적용하여 얻을수 있다. 8분나무표현은 항상 그림 13-24에서와 같이 8분공간 0, 1, 2, 3이 공간구역의 앞쪽을 형성하도록 설정된다고 가정한다.

8분나무를 현시하는 방법은 우선 8분나무의 마디들을 앞에서 뒤로 재귀절차로 순회하는것에 의해 8분나무를 보이는 구역의 4분나무에로 넘기는것이다. 다음에 보이는 면에 대한 4분나무표현은 프레임완충기에 적재된다. 그림 13-25에서는 공간구역안의 8분공간과 보기면에서의 대응하는 4분구들을 보여 주었다. 4분구 0에는 8분공간 0과 4가 관계된다. 4분구 1의 색값은 8분공간 1과 5의 면들로부터 발생된다.

공간에서의 8분공간들

보기면에 대한 4분구들

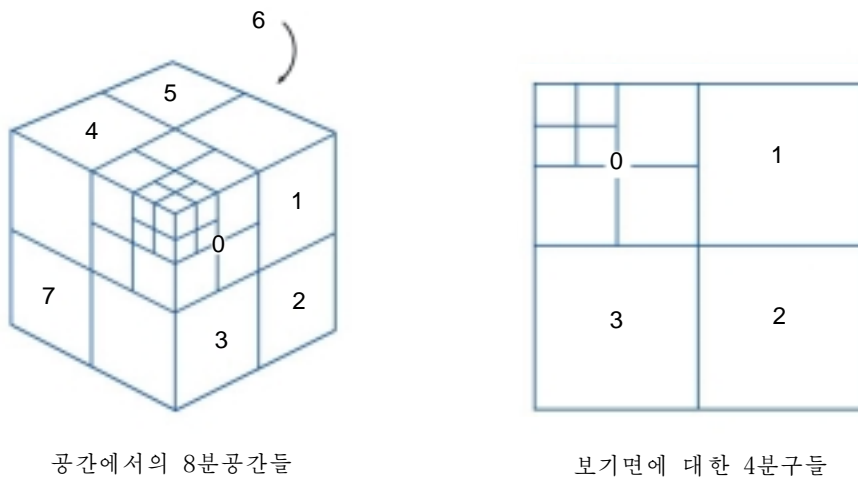



그림 13-25. 공간구역의 8분공간나누기와 대응하는 4분구평면

8분나무마디들의 재귀적인 처리를 다음의 절차에서 설명한다. 그것은 8분나무표현을 받아 들이고 공간구역안의 보이는 면들에 대한 4분나무표현을 만든다. 대부분의 경우 4분구에 대한 정확한 색값을 결정하기 위하여 앞쪽과 뒤쪽의 8분공간들을 다같이 고찰하여야 한다. 그러나 앞쪽 8분공간이 어떤 색갈로 동질 채워 지면 뒤쪽 8분공간을 처리하지 않는다. 이질공간구역에 대하여 절차는 재귀적으로 호출되며 이질8분공간의 아들과 새로 만들어 지는 4분나무마디를 새로운 인수로서 넘긴다. 앞쪽이 비여 있으면 뒤쪽 8분공간의 아들을 처리하는것만 필요하다. 그렇지 않으면 뒤쪽 8분공간과 앞쪽 8분공간에 대하여 하나씩 두개의 재귀호출이 이루어 진다.

```
typedef enum { SOLID, MIXED } Status;

#define EMPTY -1

typedef struct tOctree {
  int id;
  Status status;
  union {
    int color;
    struct tOctree * children[8];
  } data;
} Octree;

typedef struct tQuadtree {
  int id;
  Status status;
  union {
    int color;
    struct tQuadtree * children[4];
  } data;
} Quadtree;

int nQuadtree = 0;

void octreeToQuadtree (Octree * oTree, Quadtree * qTree)
{
  Octree * front, * back;
  Quadtree * newQuadtree;
  int i, j;

  if (oTree->status == SOLID) {
    qTree->status = SOLID;
    qTree->data.color = oTree->data.color;
    return;
  }
  qTree->status = MIXED;
  /* Fill in each quad of the quadtree */
```

```
    for (i=0; i<4; i++) {
      front = oTree->data.children[i];
      back = oTree->data.children[i+4];
      newQuadtree = (Quadtree *) malloc (sizeof (Quadtree));
      newQuadtree->id = nQuadtree++;
      newQuadtree->status = SOLID;
      qTree->data.children[i] = newQuadtree;

      if (front->status == SOLID)
        if (front->data.color != EMPTY)
          qTree->data.children[i]->data.color = front->data.color;
        else
          if (back->status == SOLID)
            if (back->data.color != EMPTY)
              qTree->data.children[i]->data.color = back->data.color;
            else
              qTree->data.children[i]->data.color = EMPTY;
          else { /* back node is mixed */
            newQuadtree->status = MIXED;
            octreeToQuadtree (back, newQuadtree);
          }
      else { /* front node is mixed */
        newQuadtree->status = MIXED;
        octreeToQuadtree (back, newQuadtree);
        octreeToQuadtree (front, newQuadtree);
      }
    }
  }
```

10절. 광선투사방법

그림 13-26에서와 같이 장면을 따라 보기면의 화소위치로부터의 조준선을 고찰하면 장면안의 어느 물체가 이 선과 사귀는가를 결정할수 있다. 모든 광선-면사귐점들을 계산한후 사귐점이 화소에 제일 가까운 면으로 보이는 면을 식별한다. 이 보임성검출계획은 10장 15절에서 도입된 광선투사절차를 리용한다. 보임성검출도구로서의 광선투사는 광선의 경로를 추적하는 기하광학방법에 기초한다. 장면안에는 무한개의 광선이 있으며 화소위치들을 통과하는 광선들에만 흥미가 있기때문에 장면을 따라 화소로부터 뒤쪽으로 광선경로를 추적할수 있다. 광선투사방법은 곡면 특히 구를 가지는 장면들에 대한 효과적인 보임성검출방법이다.

광선투사를 깊이완충기방법(13장 3절)의 변종으로 생각할수 있다. 깊이완충기알고리듬에서는 면들을 한번에 하나씩 처리하며 면의 모든 투영점에 대하여 깊이값을 계산한다. 계산된 면의 깊이는 다음에 매 화소에서 보이는 면을 결정하기 위하여 앞서 기억된 깊이와 비교된다. 광선투사에서는 화소들을 한번에 하나씩 처리하며 모든 면들에 대하여 화소에로의 투영경로를 따라 깊이를 계산한다.

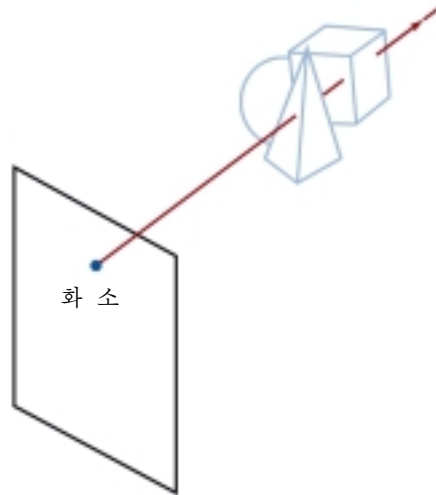



그림 13-26. 장면을 따라 화소 위치로부터의 조준선을 따르는 광선

광선투사는 대역적인 반사와 굴절에 대한 장면안의 다중물체들의 영향을 골라 내기 위하여 다중광선경로를 추적하는 광선추적알고리듬(14장 6절)의 특수경우이다. 광선투사에 의하여 매개 화소로부터 제일 가까운 물체까지 광선을 추적할뿐이다. 효률적인 광선-면사귐점계산이 일반적인 물체 특히 구에 대하여 개발되였으며 이 사귐점방법들을 14장에서 자세히 설명한다.

11절. 곡면

곡면을 가지는 물체에 대한 보임성을 결정하는 효과적인 방법에는 광선투사와 8분나무방법이 있다. 광선투사에서는 사귐점을 계산하고 화소광선을 따라 제일 작은 사귐점거리를 찾아낸다. 물체들의 8분나무표현이 입력될 때에는 보이는 모든 면들이 동일한 처리절차에 의하여 식별된다. 각이한 종류의 곡면들에 대하여 적용할수 있는 특별한 고찰방법들은 없다.

곡면은 또한 평면, 다각형면들의 모임으로 근사화할수 있다. 면들의 목록에서 매개 곡면을 다각형그물로 교체하고 앞에서 설명한 다른 보이지 않는 면검출방법들중 하나를 리용한다. 구와 같은 일부 물체들에 대하여 광선투사와 곡면의 방정식을 리용하는것은 더 정밀할뿐아니라 더 효률적일수 있다.

곡면표현

곡면은 $f(x, y, z)=0$ 형식의 음함수적인 방정식 또는 보조변수표현(부록 8)에 의하여 표현할수 있다. 실례로 스플라인곡면은 표준적으로 보조변수방정식에 의하여 표현된다. 일부 경우 양함수적인 곡면방정식을 실례로 xy바닥면우에서의 높이함수

$$z = f(x, y)$$

로 얻는것은 쓸모 있다. 구, 타원체, 원기둥, 원추와 같은 많은 흥미 있는 물체들은 2차표현을 가진다. 이 곡면들은 일반적으로 분자구조, 굴대베아링, 고리, 굴대를 모형화하는데 리용된다.

주사선 및 광선투사알고리듬들에서는 주사선 또는 화소광선과의 사귐점에서 곡면의 방정식을 풀기 위하여 수값근사기술을 리용한다. 일반적으로 리용되는 물체들에 대한 곡면방정식을 풀기 위하여 병렬계산 및 고속하드웨어계산을 비롯한 여러가지 기법들이 개발되였다.

면의 등고선도

수학, 자연과학, 공학 기타 분야들의 많은 응용에서는 곡면함수를 곡면의 형태를 보여 주는 등고선들의 모임으로 현시하는것이 편리하다. 곡면은 방정식으로 표현할수도 있고 또는 높이자료나 인구밀도자료와 같은 자료표로 표현할수 있다. 양함수적인 함수표현에 의하여 보이는 면의 등고선들을 표시하고 곡면의 보이는 부분들에 의하여 감추어 지는 부분등고선들을 없앨수 있다.

함수곡면의 xy도면을 얻기 위하여 곡면표현을

$$y = f(x, z) \tag{13-8}$$

의 형식으로 쓴다. xy평면에서의 곡선은 일부 선택된 범위안의 z값에 대하여 지적된 간격 z를 리용하여 표시될수 있다. 제일 큰 z값에서 시작하여 곡면들을 앞에서 뒤로 표시하고 보이지 않는 부분들을 없앤다. 함수에 대한 xy범위를 xy화소화면범위에로 넘기는것에 의해 곡선부분을 화면에 그린다. 다음에 x에서 단위걸음을 취하며 주어 진 z값에 대하여 식 13-8로부터 매 x값에 대한 대응하는 y값을 결정한다.

곡면에서 보이는 곡선부분을 식별하는 한가지 방법은 화면의 화소 x자리표에 대하여 앞서 계산된 y_{min}과 y_{max}값들의 목록을 유지하는것이다. 한 화소 x위치로부터 다음위치로 걸을 때 계산된 y값을 다음화소에 대하여 기억된 범위 y_{min} 및 y_{max}에 대하여 검사한다. $y_{min} \le y \le y_{max}$이면 곡면의 그 점은 보이지 않으며 그것을 표시하지 않는다. 그러나 계산된 값이 그 화소에 대한 기억된 y범위밖에 있으면 그 점은 보인다. 그러면 그 점을 표시하고 그 화소에 대한 범위를 재설정한다. xz 또는 yz평면에 등고선도를 투영할 때에도 류사한 절차들을 리용할수 있다. 그림 13-27에서는 색부호화된 등고선을 가지는 면등고선도의 실례를 보여 주었다.

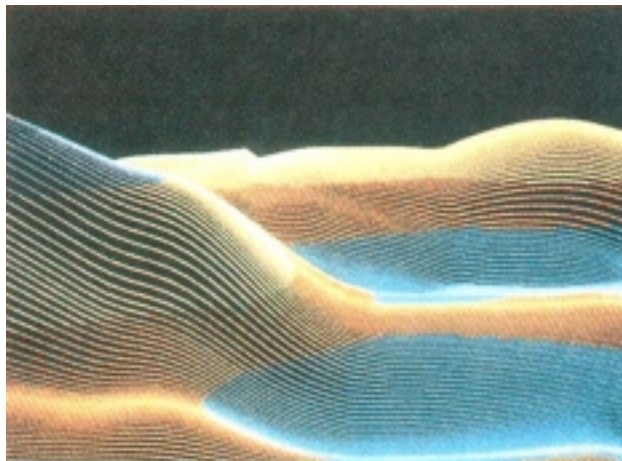



그림 13-27. 색부호화된 면등고선도

류사한 방법들은 등값면의 선들을 결정하는것에 의해 자료점들의 불련속모임에 리용될수 있다. 실례로 xy값의 $n_x \times n_y$격자에 대하여 z값들이 불련속모임이면 10장 21절에서 설명한 등고선방법을 리용하여 면우에서 z가 일정한 선경로를 결정할수 있다. 다음에 매개 선택된 등고선은 보기면에 투영될수 있으며 직선토막에 의하여 현시될수 있다. 다시 선들은 앞뒤깊이순서로 현시장치에 그어 질수 있으며 앞서 그어 진 등고선(보이는)의 뒤를 통과하는 등고선부분을 없앤다.

12절. 선그물구조방법

물체의 륜곽선만이 현시될 때 보임성검사는 면의 변에 적용된다. 보이는 부분변들은 현시되며 보이지 않는 부분변들은 없어 지거나 보이는 변과 다르게 현시될수 있다. 실례로 보이지 않는 변들을 파선으로 그을수 있으며 또는 선들의 세기를 보기면으로부터의 거리의 선형함수로 감소시키는 깊이삽입을 리용할수 있다. 물체의 변두리들의 보임성을 결정하는 절차를 선그물구조보임성방법(wireframe-visibility method)이라고 한다. 이 방법을 보이는 선의 검출방법(visible-line detection method) 또는 보이지 않는 선의 검출방법(hidden-line detection method)이라고도 한다. 선그물구조보임성절차들에 고유한 방법들이 있는데 앞절들에서 설명된 일부 보이는 면검출방법들도 변두리보임성검사에 리용할수 있다.

장면안의 보이는 선을 식별하는 직접방법은 매개 선을 매개 면과 비교하는것이다. 여기서의 처리는 지금 선의 어느 부분이 면에 의하여 감추어 지는가를 결정하는것을 제외하고 선을 임의의 창문형태에 대하여 자르는것과 류사하다. 매개 선에 대하여 깊이값은 어느 선분이 보이지 않는가를 결정하기 위하여 면과 비교된다. 실제적으로 매개 자리표위치를 검사함이 없이 보이지 않는 선분들을 식별하는데 밀착성방법을 리용할수 있다. 그림 13-28 ㄱ에서와 같이 면경계의 투영과의 선의 두 사귐점들이 다같이 그 점들에서 면보다 더 큰 깊이를 가지면 사귐점들사이의 선분은 완전히 숨겨진다. 이것은 장면에서 보통 경우이지만 서로서로 사귀는 선과 면들이 있을수 있다. 선이 한 경계사귐점에서 더 큰 깊이를 가지고 다른 경계사귐점에서 면보다 더 작은 깊이를 가질 때 선은 그림 13-28 ㄴ에서와 같이 면내부를 침투하여야 한다. 이 경우에 면의 방정식을 리용하여 선과 면의 사귐점을 계산하고 보이는 부분만을 현시한다.

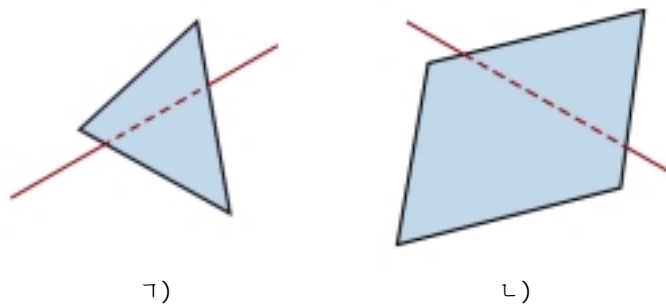



그림 13-28. 면의 뒤를 통과하는 선(ㄱ)과 면을 침투하는 선(ㄴ)에 대한 보이지 않는 부분선(파선)

일부 보이지 않는 면검출방법들은 선그물구조보임성검사에 쉽게 적용된다. 뒤면방법을 리용하여 물체의 모든 뒤면들을 식별하고 보이는 면에 대한 경계들만을 현시할수 있다. 깊이에 따르는 정렬에 의하여 면들은 면내부는 배경색, 경계는 전경색이 되도록 재생완충기에 색칠될수 있다. 면들을 뒤에서부터 앞으로 처리하는것에 의해 보이지 않는 선들은 더 가까운 면에 의해 지워 진다. 구역부분분할방법은 보이는 면의 경계만을 현시함으로써 보이지 않는 선의 소거에 적응될수 있다. 주사선방법은 주사선을 따라 보이는 면의 경계와 일치하는 점들을 설정함으로써 보이는 선을 현시하는데 리용될수 있다. 주사변환을 리용하는 임의의 보이는 면검출방법은 류사한 방식으로 변두리보임성검출방법에로 수정될수 있다.

13절. 보임성검출함수

3차원도형처리패키지들은 뒤면방법과 깊이완충기방법을 비롯하여 보이는 면을 검출하는 여러가지 절차들을 준다. backFace 또는 depthBuffer와 같은 절차의 이름에 의하여 개개의 함수를 실시할수 있다.

GKS 및 PHIGS와 같은 일반프로그람작성표준들에서 보임성방법은 실현에 관계된다. 사용가능한 방법들은 설치하는데 따라 표에 기입되며 개개 보임성검출방법은 보이지 않는 선-보이지 않는 면소거(hidden-line-hidden-surface-removal, HLHSR)함수

```
setHLHSRidentifier (visibilityFunctionIndex)
```

에 의하여 선택된다. 파라메터 visibilityFunctionIndex에는 후에 지적되는 출구기초요소들에 적용될 보임성방법을 식별하는 옹근수코드가 할당된다.

요약

여기서는 이 장에서 설명한 보임성검출방법들의 요약과 그것들의 유효성비교를 준다. 초기화면화에서 뒤면검출은 앞으로의 보임성검사들에서 많은 다각형들을 없애는데 빠르고도 효과적인 방법이다. 한개의 볼록다면체에서 뒤면검출은 모든 보이지 않는 면들을 없애지만 일반적으로 뒤면검출은 모든 보이지 않는 면들을 완전히 식별할수 없다. 다른 더 복잡한 보임성검출방법들은 보이는 면들의 목록을 정확히 만들것이다.

보이는 면들을 식별하는 빠르면서도 간단한 방법은 깊이완충기(또는 z완충기)방법이다. 이 절차는 2개의 완충기를 요구하는데 화소세기들에 대하여 한개의 완충기를 요구하고 보기면의 매 화소에 대하여 보이는 면의 깊이를 위해 한개의 완충기를 요구한다. 면들의 깊이를 계산하기 위하여 장면안의 매개 면을 훑는데 고속증분방법들이 리용된다. 매개 면이 처리될 때 두개의 완충기는 갱신된다. 깊이완충기방법을 개선한것은 A완충기방법이며 이 방법은 경계허상이 제거된 투명한 면들을 현시하기 위하여 추가적인 정보를 제공한다. 기타 보이는 면검출방법들에는 주사선방법, 깊이에 따르는 정렬방법(화가의 알고리듬), BSP나무방법, 구역부분분할방법, 8분나무방법, 광선투사방법이 있다.

보임성검출방법들은 3차원의 선화를 현시하는데도 리용된다. 곡면에 대한 등고선면을 현시할수 있다. 다면체의 선그물구조를 위하여 보기면으로부터 보이는 장면안의 면들의 부분변들을 조사한다. 보이는 면검출방법의 유효성은 개개 응용의 특성들에 관계된다. 장면안의 면들이 z방향에서 아주 작은 깊이겹침이 있도록 퍼지면 깊이에 따르는 정렬방법이나 BSP나무방법이 제일 좋은 선택이다. 수평방향으로 상당히 잘 잘라지는 면들을 가지는 장면들에서 주사선방법이나 구역부분분할방법은 보이는 면들을 찾아 내는데 능률적으로 리용될수 있다.

일반적으로 깊이에 따르는 정렬방법이나 BSP나무방법은 면들이 불과 몇개 안되는 장면들에 대하여 대단히 효과적이다. 왜냐하면 이 장면들에는 깊이에서 겹치는 면들이 얼마 없기때문이다. 주사선방법 역시 장면에 적은 개수의 면들이 있을 때 잘 수행된다. 주사선방법이나 깊이에 따르는 정렬방법, BSP나무방법은 수천개정도까지의 다각형면들을 가지는 장면들에 효과적으로 리용될수 있다. 몇천개이상의 면들을 가지는 장면들에 대하여 깊이완충기방법이나 8분나무방법이 제일 잘 수행된다. 깊이완

충기방법은 장면안의 면들의 개수에 무관계하게 거의 일정한 처리시간을 가진다. 왜냐하면 장면안의 면들의 개수가 증가할 때 면구역들의 크기는 감소하기때문이다. 따라서 깊이완충기방법은 간단한 장면들에서 상대적으로 저성능을 보여 주고 복잡한 장면들에서 상대적으로 고성능을 보여 준다. BSP나무는 서로 다른 보기참조점들을 리용하며 다중보임상을 만들려고 할 때 쓸모 있다.

8분나무표현이 체계에서 리용될 때 보이지 않는 면소거처리는 빠르고도 간단하다. 처리에서는 옹근수의 더하기와 덜기만이 리용되며 정렬이나 사귐계산은 하지 않아도 된다. 8분나무의 다른 우점은 평면이상의것을 기억하는것이다. 물체의 전체 립체구역을 현시할수 있으며 8분표현은 전체의 자름면도를 얻는데 쓸모 있다. 장면에 곡면표현들이 있으면 장면의 보이는 부분들을 식별하는데 8분나무나 광선투사방법들을 리용한다. 광선투사방법은 광선추적알고리듬의 집적부분이며 그것은 장면을 대역조명효과로 현시할수 있게 한다.

서로 다른 보이는 면검출방법들을 여러가지 방식으로 결합하고 실현할수 있다. 게다가 보임성검출알고리듬들은 흔히 하드웨어에서 실현되며 병렬처리를 리용하는 특별한 체계들은 이 방법들의 효률을 높이는데 리용된다. 비행모의기의 동화식보임상을 만드는데서와 같이 처리속도가 특별히 중요할 때 특별한 하드웨어체계들이 리용된다.

참고문헌

보임성알고리듬들에 대한 추가적인 정보원천들은 Elber와 Cohen(1990), Franklin과 Kankanhalli(1990), Glassner(1990), Naylor, Amanatides, Thibault(1990), Segal(1990)에서 보시오.

련습문제

13-1. 서로 다르게 색칠된 면들을 가지는 볼록다면체의 보이는 면들을 모두 식별하는 절차를 뒤면검출방법에 기초하여 개발하시오. 물체는 *xy*평면이 보기면인 오른손보기자리표계에서 정의된다고 가정하시오.

13-2. 볼록다면체의 보이는 면들을 보기 위하여 정투영을 리용한 뒤면검출절차를 완성하시오. 물체의 모든 부분들은 보기면의 앞에 있다고 가정하고 현시를 위하여 화면보임창에로 넘기시오.

13-3. 볼록다면체의 보이는 면들을 보기 위하여 원근투영을 리용한 뒤면검출절차를 완성하시오. 물체의 모든 부분들은 보기면의 앞에 있다고 가정하고 현시를 위하여 화면보임창에로 넘기시오.

13-4. 볼록다면체의 동화상을 만드는 프로그람을 쓰시오. 물체는 그것을 지나며 보기면에 평행인 축에 대하여 증분적으로 회전한다. 물체는 완전히 보기면의 앞에 있다고 가정하시오. 물체를 보기면에 련속적으로 넘기기 위하여 정투영을 리용하시오.

13-5. 주어 진 다면체의 보이는 면들을 현시하는 깊이완충기방법을 완성하시오. 깊이완충기에 대한 기억기요구는 현시할 물체의 정의로부터 어떻게 결정할수 있는가?

13-6. 임의의 개수의 다면체들을 포함하는 장면에서 보이는 면들을 현시하는 깊이완충기방법을 완성하시오. 장면안의 여러가지 물체들을 기억 및 처리하기 위한 효률적인 방법들을 설정하시오.

13-7. 불투명한 면과 투명한 면을 다같이 포함하는 장면을 현시하는 A완충기알고리듬을 완성하시오. 알고리듬은 경계허상을 제거하기 위하여 선택적으로 확장할수 있다.

13-8. 주어 진 다면체의 보이는 면들을 현시하는 주사선알고리듬을 완성하는 프로그람을 개발하시오. 물체의 정의를 기억하기 위하여 다각형표와 변표를 리용하고 주사선을 따라 가면서 그리고 주사선들사이에서 점들을 계산하기 위하여 밀착방법을 리용하시오.

13-9. 여러가지 다면체들을 포함하는 장면에 대하여 주사선알고리듬을 완성하는 프로그람을 쓰시오. 물체의 정의를 기억하기 위하여 다각형표와 변표를 리용하고 주사선을 따라 가면서 그리고 주사선들사이에서 점들을 계산하기 위하여 밀착방법을 리용하시오.

13-10. 화가의 알고리듬을 리용하여 볼록다면체의 보이는 면들을 현시하는 프로그람을 작성하시오. 즉 면들은 깊이에 따라 정렬되며 뒤에서부터 앞으로 나가면서 화면에 색칠된다.

13-11. 평면들을 가지는 임의로 주어진 물체의 보이는 면들을 현시하기 위하여 깊이에 따르는 정렬방법을 리용하는 프로그람을 쓰시오.

13-12. 여러가지 다면체들을 포함하는 장면에서 보이는 면들을 현시하는 깊이정렬프로그람을 개발하시오.

13-13. BSP나무방법을 리용하여 볼록다면체의 보이는 면들을 현시하는 프로그람을 쓰시오.

13-14. 구역부분분할알고리듬의 검사 3에서 론의한 두개의 방법들이 다른 모든 면들을 덮어 감추는 둘러 싸는 면을 정확히 식별하지 못하고 실패하는 실례들을 주시오.

13-15. 직4각형구역에 대하여 주어 진 평면이 둘러 싸는 면인가, 겹치는 면인가, 내부면인가, 외부면인가를 결정하기 위하여 검사하는 알고리듬을 개발하시오.

13-16. 4분나무원소들의 값을 결정하는데 구역부분분할검사들을 적용하여 물체의 보이는 면들에 대한 4분나무표현을 만드는 알고리듬을 개발하시오.

13-17. 현시를 위하여 물체의 주어 진 4분나무표현을 프레임완충기에 적재하는 알고리듬을 작성하시오.

13-18. 보이지 않는 면들이 제거되도록 임의의 8분나무표현을 현시하는 프로그람을 체계에서 쓰시오.

13-19. 광선투사방법을 리용하여 한개의 구를 보는 알고리듬을 창안하시오.

13-20. 경계허상제거방법을 여러가지 보이지 않는 면소거알고리듬들에 어떻게 병합할수 있는가를 론의하시오.

13-21. 주어 진 곡면함수에 대한 면등고선도면을 만드는 루틴을 쓰시오.

13-22. 장면안의 매선을 매면과 비교하여 장면안에서 보이는 부분선들을 검출하는 알고리듬을 개발하시오.

13-23. 이 장에서 론의한 여러가지 보이는 면검출방법들에 의하여 어떻게 선그물구조현시를 만들겠는가를 론의하시오.

13-24. 물체의 보이지 않는 변두리들을 파선으로 그려 다면체의 선그물구조현시를 만드는 절차를 설정하시오.

14장. 조명모형 및 면실감처리

장면을 현실감 있게 현시하려면 물체들을 원근투영하고 보이는 면들에 자연적인 조명효과를 적용하여야 한다. **조명모형**(illumination model, lighting model)은 물체면우의 주어 진 점에서 보게 되는 빛의 세기를 계산하는데 리용한다. 조명모형을 명암모형(shading model)이라고도 한다. **면실감처리알고리듬**(surface rendering algorithm)에서는 조명모형에서의 빛세기계산을 리용하여 장면안의 여러 면들의 모든 투영된 화소위치들에 대한 세기를 결정한다. 면실감처리는 보이는 면우의 모든 점에 대하여 조명모형을 적용하거나 또는 일부 점에 대하여 조명모형계산을 진행하고 나머지 점들의 빛세기는 보간하는 방법으로 할수도 있다. 일반적으로 주사선알고리듬과 화상공간알고리듬들에서는 보간방법을 리용하며 광선추적알고리듬들에서는 조명모형을 매개 화소위치에 적용한다. 면실감처리절차를 면명암방법(surface shading method)이라고도 한다. 혼란을 피하기 위하여 용어 조명모형은 면우의 한 점에서 빛세기를 계산하는 모형을 의미하며 용어 면실감처리는 장면안의 면들이 투영된 모든 화소위치들에서 세기를 얻기 위하여 조명모형을 적용하는 절차를 의미하는데 리용하겠다.

콤퓨터도형처리에서 사진과 같은 감을 낸다는것은 두가지 즉 장면에서 물체들을 도형적으로 정밀하게 표현한다는것과 조명효과를 물리적으로 잘 표현한다는것을 의미한다. 조명효과에서는 빛반사, 투명도, 면의 결문양, 그림자를 고려하여야 한다.

물체에서 우리가 보게 되는 색과 조명효과를 모형화하는 과정은 물리학과 심리학의 원리들을 다같이 포함하는 복잡한 과정이다. 기본적으로 조명효과는 전자기에네르기와 물체면과의 호상작용을 고찰하는 모형에 의하여 표현된다. 빛이 우리 눈에 닿으면 장면에서 실제로 무엇이 보이는가를 결정하는 감각과정이 일어 난다. 장면에서 설정하는 물체의 류형, 광원과 다른 물체들에 대한 물체의 위치, 광원의 조건과 같은 여러가지 인자들을 물리적인 조명모형에 포함시킨다. 물체는 불투명재료로 만들어 질수도 있고 얼마간 투명할수도 있다. 물체는 광택 또는 무광택면을 가질수 있으며 여러가지 결문양을 가질수 있다. 장면에 여러가지 조명효과를 주기 위하여 광원의 형태와 색갈, 위치를 변화시킬수 있다. 면들의 광학특성에 대한 파라메터, 장면안에서 면들의 상대적인 위치, 광원의 색갈과 위치, 보기면의 위치와 방향이 주어 지면 조명모형은 면우의 개개 점으로부터 지적된 보기방향으로 투영되는 세기를 계산한다.

콤퓨터도형처리에서 조명모형은 면의 빛세기를 표현하는 물리법칙으로부터 대략적으로 유도된다. 세기계산을 최소화하기 위하여 대부분의 프로그람들은 간단화된 광도측정계산에 기초하는 경험적인 모형을 리용한다. 복사세기알고리듬과 같은 보다 정밀한 모형들은 장면안의 면과 광원들사이의 복사에네르기의 전파를 고찰하는 방법으로 빛세기를 계산한다. 다음절들에서 먼저 도형처리프로그람들에서 자주 리용되는 기본조명모형을 보고 다음에 면의 세기를 보다 정밀하게 계산하기는 하지만 시간이 오래 걸리는 방법들을 설명한다. 그리고 장면안의 보이는 면들에서의 적당한 명암을 얻는데 조명모형을 적용하는 여러가지 면실감처리알고리듬들을 연구한다.

1절. 광원

불투명한 비발광물체를 볼 때 물체면에서 반사되는 빛을 본다. 전체 반사빛은 장면안의 광원과 다른 반사면들로부터 나오는 빛이 반사되여 결합된것이다(그림 14-1). 그러므로 면은 광원에 의하여 직접 로출되지 않아도 이웃한 물체들이 조명되면 보일것이다. 때때로 광원을 빛방출원(light-emitting source)이라고 하며 방안의 벽과 같은 반사면들은 빛반사원(light-reflecting source)이라고 한다. 용어 **광원**(light source)은 백열등이나 태양과 같이 복사에네르기를 방출하고 있는 물체를 의미하는데 리용

하겠다.

일반적으로 발광물체는 광원으로 될수도 있고 빛반사원으로 될수도 있다. 실례로 내부에 백열등이 있는 합성수지장갑은 장갑의 면으로부터 빛을 방출 및 반사시킨다. 장갑으로부터 방출된 빛은 다음에 근방의 다른 물체들을 조명한다.

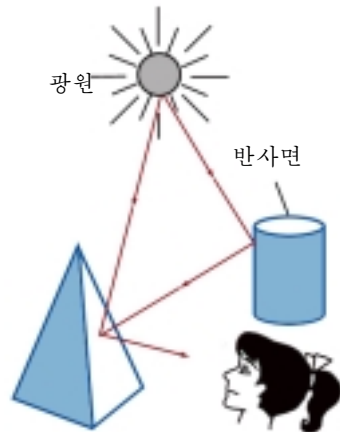



그림 14-1. 불투명한 비발광면으로부터 보이는 빛은 일반적으로 광원으로부터 나오는 빛과 다른 면들로부터 나오는 반사빛이 반사되여 결합된것이다.

광원의 제일 간단한 모형은 **점광원**(point source)이다. 그림 14-2에 보여 준바와 같이 빛은 점광원으로부터 해살모양의 경로를 따라 나온다. 광원의 이 모형은 장면안의 물체들에 비하여 크기가 작은 광원들에 대한 합리적인 근사이다. 태양과 같이 장면으로부터 충분히 먼 광원들은 점광원으로 정밀하게 모형화될수 있다. 그림 14-3의 긴 형광등과 같은 가까운 광원은 **분산광원**(distributed light source)으로 보다 정밀하게 모형화된다. 이 경우에 조명효과는 점광원에 의하여 현실감 있게 근사화될수 없다. 왜냐하면 광원의 면적이 장면안의 면들에 비하여 작지 않기때문이다. 분산광원은 광원의 면에 점들이 축적된 조명효과를 고찰하여 정밀하게 모형화할수 있다.

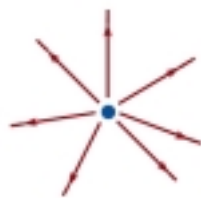



그림 14-2. 점광원으로부터 나오는 해살모양의 광선경로

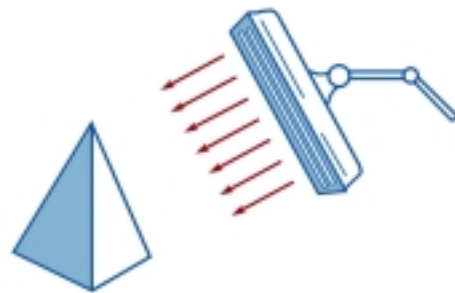



그림 14-3. 분산광원에 의하여 조명되는 물체

빛이 불투명한 물체에 입사할 때 일부는 반사되고 일부는 흡수된다. 면에 의하여 반사되는 입사빛의 량은 재질의 류형에 관계된다. 광택재질은 입사빛을 더 많이 반사시키며 무광택면은 입사빛을 더 많이 흡수한다. 한편 빛이 투명한 물체에 입사할 때에는 일부는 반사되고 일부는 재질을 통하여 투과된다.

거칠거칠한 즉 나무결모양의 면은 반사빛을 모든 방향으로 산란시키는 경향이 있다. 이 산란된 빛을 **확산반사**(diffuse reflection)빛이라고 한다. 대단히 거칠거칠한 무광택면은 주로 확산반사를 만들며 따라서 면은 모든 보기방향으로부터 같은 밝기로 나타난다. 그림 14-4에서는 면으로부터 산란되는

확산빛을 보여 주었다. 물체의 색이라고 부르는것은 확산반사된 입사빛의 색이다. 실례로 백색광원에 의하여 조명되는 푸른 물체는 백색광의 푸른 성분을 반사시키고 나머지 성분들은 모두 흡수한다. 푸른 물체를 붉은색광원하에서 보면 모든 입사빛이 흡수되기때문에 검은 색으로 나타난다.

확산반사외에 광원은 **거울반사**(specular reflection)라고 하는 광채 즉 가장 밝은 부분을 만든다. 광채효과는 무광택면보다도 광택면에서 더 많이 나타난다. 그림 14-5에서는 거울반사를 보여 주었다.

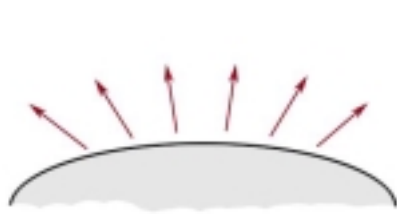



그림 14-4. 면으로부터의 확산반사

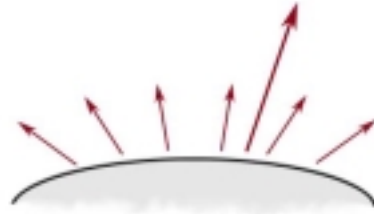



그림 14-5. 확산반사벡토르우에 덧놓여 지는 거울반사벡토르

2절. 기본조명모형

여기서는 빛세기를 계산하는 간단한 방법들을 설명한다. 이 절에서는 주어 진 점에서 면의 세기를 계산하는 간단하고 빠른 경험적인 모형들을 설명하는데 그것들은 대부분의 장면들에서 아주 좋은 결과들을 만든다. 조명계산은 면의 광학특성, 배경빛조건, 광원지적에 기초한다. 광학파라메터들은 광택, 무광택, 불투명, 투명과 같은 면특성들을 설정하는데 리용된다. 이것은 입사빛의 반사 및 흡수량을 조종한다. 모든 광원들은 점광원으로 고찰되며 자리표위치와 세기값(색)에 의하여 지적된다.

주변빛

광원에 의하여 직접 로출되지 않는 면은 이웃한 물체들이 조명되면 보일것이다. 기본조명모형에서는 장면에 대한 일반적인 밝기준위를 설정할수 있다. 이것은 **주변빛**(ambient light) 또는 배경빛(background light)이라고 하는 일정한 조명을 만들도록 여러가지 면들로부터의 반사빛들의 결합을 모형화하는 간단한 방법이다. 주변빛은 공간적인 또는 방향적인 특성이 없다. 매개 물체에 입사하는 주변빛의 량은 모든 면에 대하여 모든 방향에서 일정하다.

장면에서 주변빛의 준위를 파라메터 I_a로 설정할수 있으며 매개 면은 이 상수값에 의하여 조명된다. 매면에 입사하는 주변빛의 세기는 일정하며 보기방향과 면의 공간적인 방향에 무관계하다. 그러나 매면에서 나오는 반사빛의 세기는 면의 광학특성 즉 얼마만한 입사에네르기가 반사되며 얼마만한것이 흡수되는가 하는데 관계된다.

확산반사

주변빛에 의한 반사는 대역적인 확산조명효과의 근사이다. 확산반사는 장면안의 매면에서 일정하며 보기방향에 독립이다. 매면에 대하여 입사빛이 확산반사된 비률은 확산반사결수(diffuse-reflection coefficient) 또는 확산반사률(diffuse reflectivity) k_d에 의하여 설정할수 있다. 파라메터 k_d에는 면이 가지는 반사특성에 따라 구간 0∼1사이의 상수값이 할당된다. 대부분의 입사빛을 반사시키는 면을 요구하면 k_d의 값을 1에 가깝게 설정한다. 이것은 반사빛의 세기가 입사빛의 세기에 가까운 밝은 면을 만든다. 대부분의 입사빛을 흡수하는 면에 대해서는 반사률을 0에 가까운 값으로 설정

한다. 실제로 파라메터 k_d는 면색갈의 함수이지만 당분간 k_d를 상수라고 가정하겠다.

면이 주변빛만을 받는다면 면우의 임의의 점에서 확산반사의 세기는

$$I_{\text{ambdiff}} = k_d I_a \tag{14-1}$$

과 같이 표현할수 있다. 주변빛은 매개 면에 대하여 흥미 없는 단조로운 명암을 만들기때문에(그림 14-9 ㄴ) 장면은 주변빛 하나만으로는 거의 실감처리되지 않는다. 장면에는 적어도 하나의 광원, 흔히 보기위치에 있는 점광원이 포함된다.

점광원에 의한 조명의 확산반사를 류사한 방법으로 모형화할수 있다. 즉 면에서 나오는 확산반사빛은 모든 방향에서 같은 세기로 산란되며 보기방향에 독립이라고 가정한다. 이러한 면을 때때로 리상적인 확산반사면(ideal diffuse reflector)이라고 한다. 그것은 또한 램버트반사면(Lambertian reflector)이라고도 한다. 왜냐하면 면우의 임의의 점에서 복사되는 빛에네르기는 램버트의 코시누스법칙(Lambert's cosine law)에 의하여 결정되기때문이다. 이 법칙에 의하면 임의의 작은 면적 dA로부터 면의 법선에 대하여 임의의 방향 ϕ_N으로 복사되는 에네르기는 $\cos\phi_N$에 비례한다(그림 14-6). 그러나 빛세기는 방향 ϕ_N에 수직인 투영면적당 복사에네르기에 관계되며 그것은 $dA \cdot \cos\phi_N$이다. 그러므로 램버트반사에서 빛세기는 모든 보기방향에서 같다. 복사에네르기와 같은 광도측정개념과 용어들을 14장 7절에서 더 자세히 설명한다.

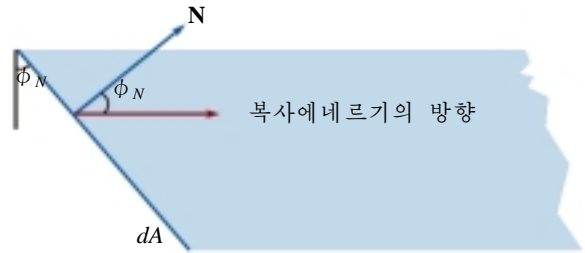



그림 14-6. 면적 dA로부터 면의 법선방향에 대하여 방향 ϕ_N으로 복사하는 에네르기

완전한 확산반사면으로부터 빛은 모든 방향에서 꼭같이 산란되지만 면의 밝기는 광원에 대한 면의 방향에 관계된다. 입사빛의 방향에 수직인 면은 입사빛의 방향에 수직이 아닌 면보다 더 밝게 나타난다. 이것은 흰색종이판 또는 매끈한 마분지를 가까운 창문에 평행으로 유지하고 판을 창문방향으로부터 떨어 져서 천천히 회전시키는 방법으로 쉽게 볼수 있다. 면의 법선과 입사빛의 방향사이의 각이 증가할 때 그림 14-7에 보여 준바와 같이 더 작은 입사빛이 면에 떨어 진다. 이 그림은 먼 거리의 광원(평행입사광선)으로부터 오는 빛방향에 대하여 서로 다른 공간방향을 가지는 두개의 등면적평면쪼각에 입사하는 광속을 보여 준다. 입사빛의 방향과 면의 법선사이의 입사각을 θ로 표시하면(그림 14-8) 빛방향에 수직인 부분면의 투영면적은 $\cos\theta$에 비례한다. 그러므로 조명량(또는 투영된 부분면을 가로질러 자르는 입사광선의 개수)은 $\cos\theta$에 관계된다. 광원으로부터 들어 오는 빛이 어떤 점에서 면에 수직이면 그 점은 완전히 조명된다. 조명각이 면의 법선으로부터 떨어 져 갈 때 점의 밝기는 떨어 진다. I_l가 점광원의 세기이면 면우의 점에 대한 확산반사식은

$$I_{l,\text{diff}} = k_d I_l \cos\theta \tag{14-2}$$

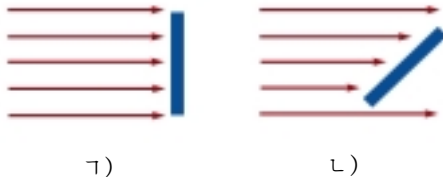



그림 14-7. 입사빛의 방향에 수직인 면(ㄱ)은 들어 오는 빛방향에 대하여 빗각인 같은 크기면(ㄴ)보다 더 많이 조명된다.

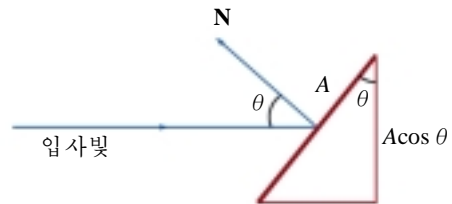



그림 14-8. 들어 오는 광선경로에 수직으로 투영되는 조명면적

와 같이 쓸수 있다. 면은 입사각이 0～90° 범위(cos θ는 이 구간에서 0～1사이에 있다.)에 있을 때에만 점광원에 의하여 조명된다. cos θ가 부일 때 광원은 면의 뒤에 있다.

N이 면의 단위법선벡토르이고 **L**이 면우의 위치로부터 점광원에로의 단위방향벡토르이면(그림 14-9) cos θ = **N**・**L**이고 점광원에 의한 조명의 확산반사식은

$$I_{l,\text{diff}} = k_d I_l (\mathbf{N} \cdot \mathbf{L}) \quad (14-3)$$

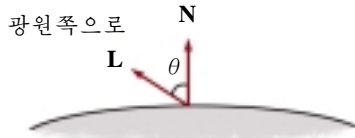



그림 14-9. 광원의 단위방향벡토르 **L**과 면의 단위법선벡토르 **N**사이의 입사각 θ

이다. 점광원에 의한 조명의 반사는 쏠림 및 원근변환이 적용되기전에 세계자리표 또는 보기자리표에서 계산된다. 이 변환들은 법선벡토르의 방향을 그것이 표현하는 면에 수직이 아니도록 변환할수 있다. 면의 법선의 고유한 방향을 유지하는 변환절차들을 11장에서 설명하였다.

그림 14-10에서는 파라메터 k_d를 0～1사이에서 변화시키면서 구면우의 위치들에 식 14-3을 적용한 실례를 보여 주었다. 면이 투영된 매개 화소위치에는 점광원의 확산반사식에 의하여 계산된 세기가 할당되였다. 이 그림에서는 다른 조명효과없이 하나의 점광원만으로 조명하였을 때의 실감처리상들을 보여 주었다. 이것은 완전히 어두운 방에서 물체를 작은 등으로 비칠 때 볼수 있는것이다. 그러나 일반적인 장면들에서는 직사광원에 의하여 만들어 지는 조명효과외에 일부 배경조명효과를 고려한다.

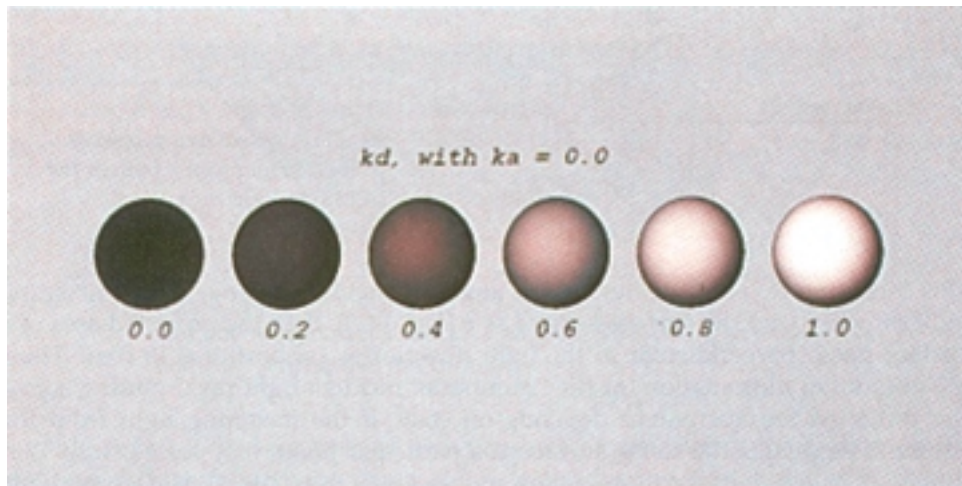



그림 14-10. 확산반사률이 $0 \leq k_d \leq 1$일 때 점광원에 의하여 조명되는 구면에서의 확산반사

전체 확산반사에 대한 표현을 얻기 위하여 주변 및 점광원세기계산을 결합할수 있다. 대부분의 도형처리프로그람들에서는 매면에 대한 주변빛세기 I_a를 수정하기 위하여 주변반사결수(ambient-reflection coefficient) k_a를 도입한다. 이것은 장면의 빛조건들을 조정하는 추가적인 파라메터를 간단히 제공한다. 파라메터 k_a를 리용하여 전체 확산반사식을

$$I_{\text{diff}} = k_a I_a + k_d I_l (\mathbf{N} \cdot \mathbf{L}) \quad (14-4)$$

와 같이 쓸수 있다. 여기서 k_a와 k_d는 다같이 면의 재질특성에 관계되며 0～1사이의 값을 가진다. 그림 14-11은 0～1사이의 파라메터 k_a와 k_d값에 대하여 식 14-4로부터 계산되는 면세기에 의하여 현시되는 구를 보여 준다.

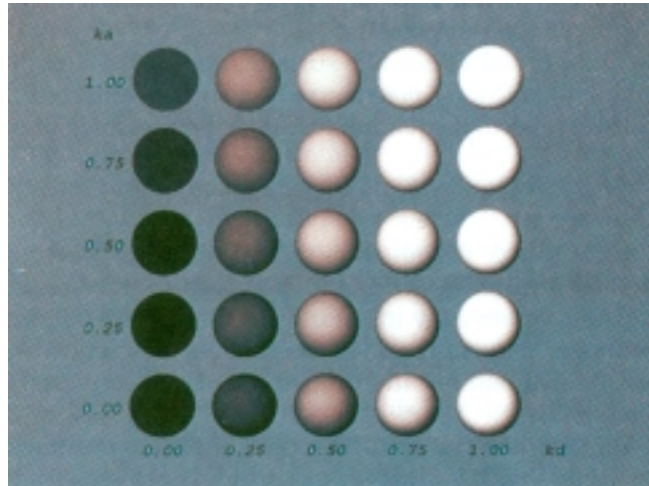



그림 14-11. k_a와 k_d가 (0, 1)구간의 값을 가질 때 주변빛과 하나의 점광원에 의하여 조명되는 구걸면에서의 확산반사

거울반사와 퐁모형

광채 나는 금속, 사과, 사람의 이마와 같은 조명되는 비침면을 볼 때 일정한 보기방향에서 광채 또는 가장 밝은 부분을 본다. 거울반사라고 하는 이 현상은 거울반사각주위의 집중구역에서 입사빛이 총체적으로 또는 거의 총제적으로 반사된 결과이다. 그림 14-12에서는 조명되는 면우의 점에서 거울반사의 방향을 보여 주었다. 거울반사각은 입사빛의 각과 같으며 두 각은 면의 단위법선벡토르 **N**의 서로 반대쪽에서 측정된다. 이 그림에서 **R**는 리상적인 거울반사방향의 단위벡토르, **L**은 점광원쪽으로 향하는 단위벡토르, **V**는 면우의 위치로부터 관찰자쪽을 가리키는 단위벡토르를 표현하는데 리용한다. 각 ϕ는 거울반사방향에 대한 보기각이다. 리상적인 반사면(완전한 거울)에 대하여 입사빛은 거울반사방향으로만 반사된다. 이 경우 벡토르 **V**와 **R**가 일치($\phi=0$)할 때에만 반사빛을 볼수 있다.

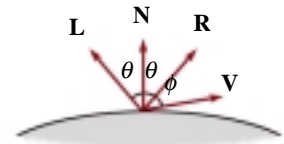



그림 14-12. 거울반사각은 입사각 θ와 같다.

리상적인 반사면과 다른 물체들은 벡토르 **R**주위의 보기위치의 유한범위에서 거울반사를 보여 준다. 광택면은 좁은 거울반사범위를 가지며 무광택면은 더 넓은 반사범위를 가진다. Phong Bui Tuong에 의하여 개발되고 **퐁의 거울반사모형**이라고 하는 거울반사범위를 계산하는 경험적인 모형은 거울반사의 세기를 $\cos^{n_s}\phi$에 비례하도록 설정한다. 각 ϕ에는 0~90°사이의 값이 할당될수 있으며 따라서 $\cos\phi$는 0~1사이에서 변한다. 거울반사파라메터 n_s에 할당되는 값은 현시하려는 면의 류형에 의하여 결정된다. n_s는 고광택면들에서 큰 값(100 또는 그이상)으로 모형화되며 보다 무광택인 면들에서는 더 작은 값(1아래)으로 모형화된다. 완전한 반사면에 대하여 n_s는 무한대이다. 분필이나 탄재블로크와 같은 거칠거칠한 면일 때 n_s에는 1에 가까운 값이 할당된다. 그림 14-13과 14-14에서는 거울반사를 볼수 있는 각범위에 대한 n_s의 효과를 보여 준다.

거울반사의 세기는 입사빛의 쏠림과 색과 같은 다른 인자들뿐아니라 면의 재질특성과 입사각에도 관계된다. 단색거울세기변화를 매면에 대한 거울반사결수 $W(\theta)$를 리용하여 근사적으로 모형화할수 있다. 그림 14-15는 몇가지 재질에 대하여 $\theta=0°$부터 $\theta=90°$까지 범위에서 $W(\theta)$의 일반적인 변화를 보여 준다. 일반적으로 입사각이 커질 때 $W(\theta)$는 커지는 경향이 있다. $\theta=90°$에서 $W(\theta)=1$

이며 모든 입사빛은 반사된다. 입사각에 의한 거울세기의 변화는 프레스넬(Fresnel)의 반사법칙에 의

그림 14-13. 파라메터 n_s에 대한 거울반사(색칠된 구역)의 모형화

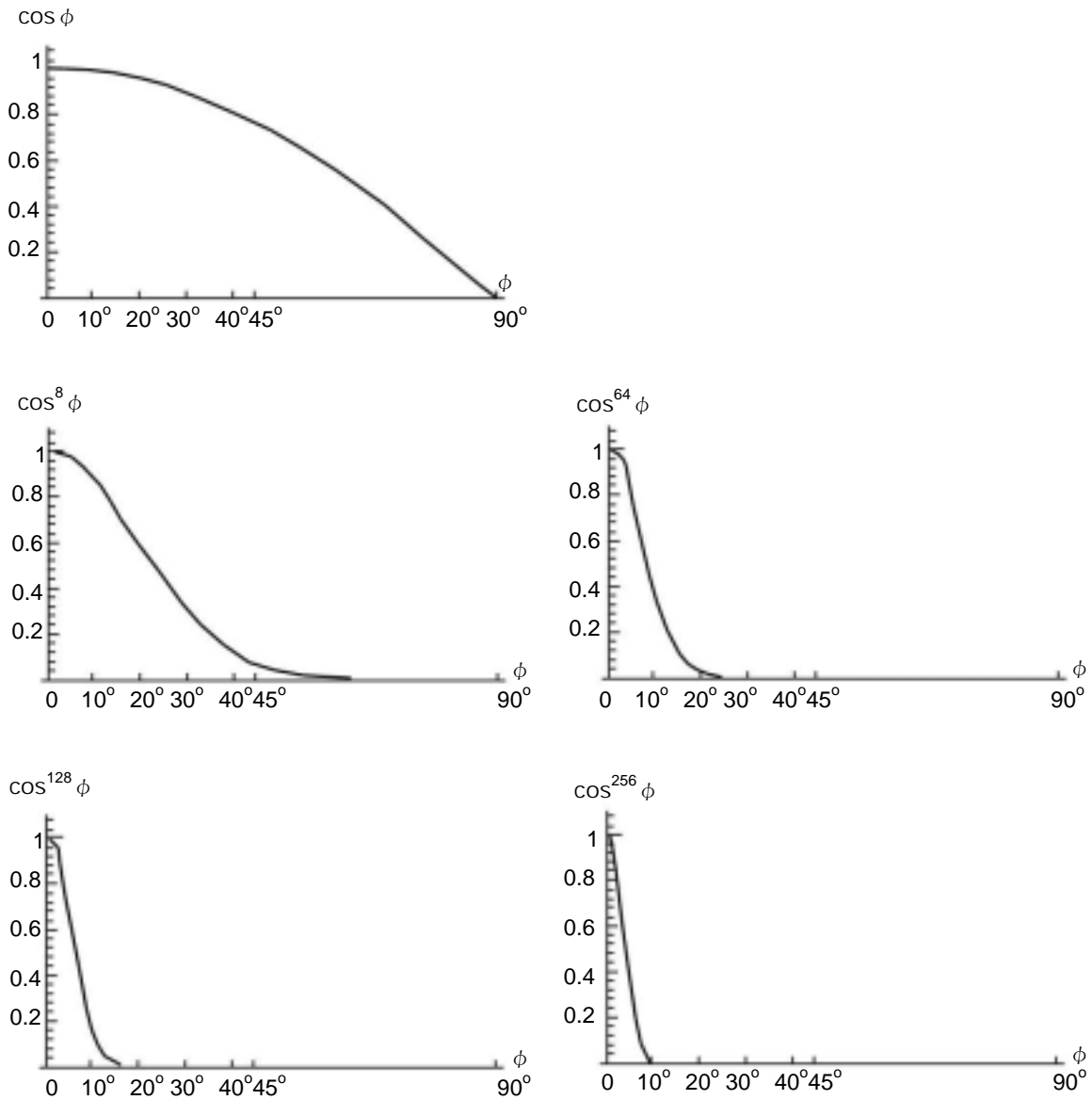



그림 14-14. 거울파라메터 n_s의 여러가지 값에 대한 $\cos^{n_s}\phi$의 곡선

하여 표현된다. 스펙트르반사함수 $W(\theta)$를 리용하여 퐁의 거울반사모형을

$$I_{\text{spec}} = W(\theta) I_l \cos^{n_s}\phi \tag{14-5}$$

와 같이 쓸수 있다. 여기서 I_l은 광원의 세기이며 ϕ는 거울반사방향 **R**에 대한 보기각이다.

그림 14-15에서 보는바와 같이 유리와 같은 투명한 재질들에서는 θ가 90°에 다가갈 때 거울반사가 극히 적다. θ =0°에서 유리면에 입사하는 빛의 약 4%가 반사된다. 그리고 θ의 대부분의 범위에서 반사세기는 입사세기의 10%보다 작다. 그러나 대부분의 불투명한 재질들에서 거울반사는 모든 입사각에 대하여 거의 일정하다. 이 경우에 $W(\theta)$를 일정한 거울반사결수 k_s로 교체하여 반사빛효과를 합리적으로 모형화할수 있다. 다음에 k_s를 매면에 대하여 0∼1사이범위의 어떤 값과 같도록 설정한다.

그림 14-15. 서로 다른 재질들에 대하여 입사각이 함수로서 거울반사결수의 조사변화

V와 **R**는 보기 및 거울반사방향에서의 단위벡토리이기때문에 cos ϕ의 값을 스칼라적 **V・R**에 의하여 계산할수 있다. 거울반사결수가 일정하다고 하면 면의 점에서의 거울반사의 세기를 계산

$$I_{\text{spec}} = k_s I_l (\mathbf{V} \cdot \mathbf{R})^{n_s} \qquad (14-6)$$

에 의하여 결정할수 있다. 이 식에서 벡토르 **R**는 벡토르 **L**과 **N**에 의하여 계산할수 있다. 그림 14-16에서 보는바와 같이 **L**의 법선벡토르방향에로의 투영은 스칼라적 **N・L**에 의하여 얻어 진다. 따라서 그림으로부터

$$\mathbf{R} + \mathbf{L} = (2\mathbf{N} \cdot \mathbf{L})\mathbf{N}$$

이며 거울반사벡토르는

$$\mathbf{R} = (2\mathbf{N} \cdot \mathbf{L})\mathbf{N} - \mathbf{L} \qquad (14-7)$$

과 같이 얻어 진다. 그림 14-17에서는 하나의 점광원에 의하여 조명될 때 k_s와 n_s의 여러가지 값에 대한 거울반사를 보여 주었다.

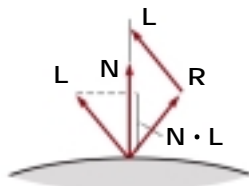



그림 14-16. 법선벡토르 **N**의 방향에로의 투영을 고찰하여 벡토르 **R**를 계산

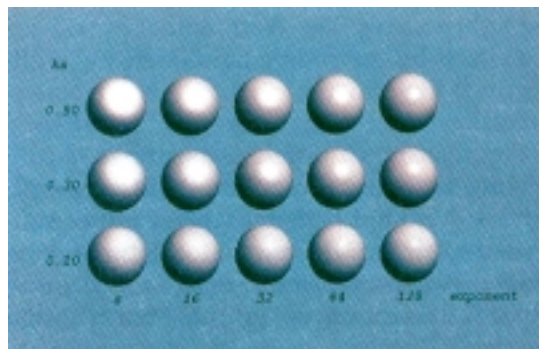



그림 14-17. 여러가지 거울파라메터 값과 하나의 광원에 대한 구면에서의 거울반사

약간 간단화된 퐁모형은 거울반사의 범위를 계산하는데 **L**과 **V**사이의 중간벡토르 **H**를 리용하여 얻는다. 퐁모형에서 **V・R**를 스칼라적 **N・H**로 바꾸면 이것은 경험적인 cos ϕ계산을 경험적인 cos α계산으로 교체한다(그림 14-18). 중간벡토르는

$$\mathbf{H} = \frac{\mathbf{L} + \mathbf{V}}{|\mathbf{L} + \mathbf{V}|} \qquad (14-8)$$

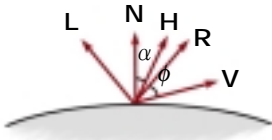



그림 14-18. **L** 과 **V** 사이의 각 2 등분선을 따르는 중간벡토르 **H**

과 같이 얻어 진다. 관찰자와 광원이 다같이 면으로부터 충분히 멀리에 있으면 **V**와 **L**은 다같이 면우에서 일정하며 따라서 **H**는 모든 면점들에 대하여 일정하다. 곡면에 대하여 **N**·**H** 는 **V**·**R**보다 더 적은 계산을 요구한다. 왜냐하면 매개 면점에서 **R**의 계산은 변수벡토르 **N**을 포함하기때문이다.

주어 진 광원과 관찰자의 위치에 대하여 벡토르 H는 보기방향에서 최대의 거울반사를 만드는 면의 방향이다. 이 리유로 하여 H는 때때로 최대광채에 대한 면의 방향이라고 한다. 또한 벡토르 V가 벡토르 L 및 R(따라서 N)와 같은 평면우에 있으면 각 α는 값 $\phi/2$를 가진다. V, L, N이 같은 평면우에 있지 않을 때 $\alpha > \phi/2$이며 세 벡토르의 공간적인 관계에 의존한다.

여러 광원의 확산반사와 거울반사의 결합

점광원이 하나일 때 조명되는 면우의 점에서 확산반사와 거울반사의 결합은

$$\begin{aligned} I &= I_{\text{diff}} + I_{\text{spec}} \\ &= k_a I_a + k_d I_l (\mathbf{N}\cdot\mathbf{L}) + k_s I_l (\mathbf{N}\cdot\mathbf{H})^{n_s} \end{aligned} \qquad (14-9)$$

와 같이 모형화할수 있다. 그림 14-19에서는 식 14-9의 여러가지 항들에 의하여 만들어 진 면의 조명효과를 보여 주었다. 장면에 하나이상의 점광원을 놓으면 임의의 면점에서의 빛반사는 개별적인 광원들의 빛반사를 합하여 얻는다.

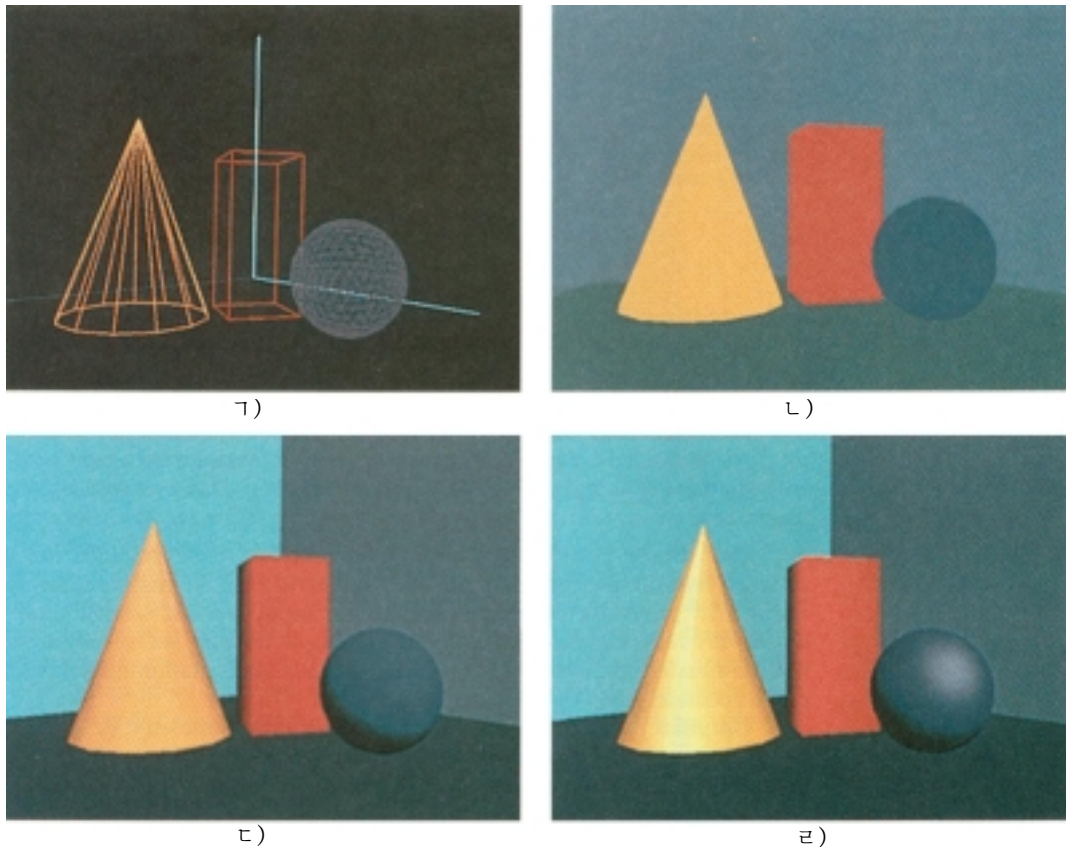



그림 14-19. 선그물구조장면 ㄱ는 ㄴ에서 주변빛에 의해서만 현시되며 매 물체의 면에는 서로 다른 색이 할당된다. 주변빛과 모든 면에 대하여 $k_s = 0$인 단일광원으로 인한 확산반사를 리용하여 ㄷ에 보여 준 조명효과를 얻는다. 주변빛과 단일광원으로 인한 확산 및 거울반사를 다같이 리용하여 ㄹ에 보여 준 조명효과를 얻는다.

$$I = k_a I_a + \sum_{i=1}^{n} I_{li} [k_d (\mathbf{N} \cdot \mathbf{L}_i) + k_s (\mathbf{N} \cdot \mathbf{H}_i)^{n_s}] \qquad (14-10)$$

임의의 화소세기가 최대허용값을 넘지 않는다는것을 보증하기 위하여 일부 류형의 정규화절차를 적용할수 있다. 간단한 방법은 세기식의 매 항에 대하여 최대크기를 설정하는것이다. 임의의 계산된 항이 최대값을 넘으면 그것을 간단히 최대값으로 설정한다. 세기넘침을 보상하는 다른 방법은 개별적인 항들을 최대항의 크기로 나누어 정규화한다. 보다 복잡한 절차는 먼저 장면에 대하여 모든 화소세기들을 계산하고 다음에 계산된 세기들을 허용세기범위에로 비례변환하는것이다.

완모형

이제까지 점광원만을 고찰하였다. **완모형**(Warn model)은 빛세기를 서로 다른 방향에서 조종하여 촬영장조명효과를 모의하는 방법이다.

광원은 면의 점에 대한 퐁모형을 리용하여 반사면우의 점들로 모형화된다. 그러면 서로 다른 방향에서의 세기는 퐁지수에 대한 값을 선택하여 조종한다. 게다가 《차광판》과 국부조명과 같은 촬영가가 리용하는 빛조종은 완모형에서 모의할수 있다. 갓은 광원에 의하여 여러가지 방향으로 방출되는 빛량을 조종하는데 리용된다. x, y, z의 매 방향에 대하여 두개의 갓이 있다. 국부조명기구는 정점이 점광원위치에 있는 원추안에서 방출되는 빛량을 조종하는데 리용된다. 완모형은 PHIGS+에서 실현되며 그림 14-20에서는 이 모형에 의하여 만든 조명효과를 보여 주었다.

그림 14-20. Chevrolet Camaro 를 조명하기 위하여 5 개의 광원을 리용하는 완모형에 의하여 만들어 진 촬영장조명효과

세기감쇠

점광원으로부터 복사에네르기가 공간을 따라 움직일 때 그의 진폭은 인자 $1/d^2$에 의하여 감쇠된다. 여기서 d는 빛이 움직인 거리이다. 이것은 광원에 가까운 면(d가 작은 경우)은 먼 거리의 면(d가 큰 경우)보다 광원으로부터 더 높은 입사세기를 받는다는것을 의미한다. 따라서 현실감 있는 조명효과를 만들기 위해서는 조명모형에서 이 세기감쇠를 고려하여야 한다. 그렇지 않으면 면들이 광원으로부터 얼마나 멀리 있던지간에 모든 면들을 동일한 세기로 조명한다. 동일한 광학파라메터를 가지는 두개의 평행면이 겹치면 그것들은 서로 구별할수 없다. 두 면은 한개 면으로 현시될것이다.

그러나 간단한 점광원모형은 세기를 감쇠시키는데 인자 $1/d^2$을 리용하여도 언제나 현실감 있는 그림을 만드는것은 아니다. 인자 $1/d^2$은 d가 작을 때 너무 큰 세기변화를 만들며 d가 클 때 대단히 작은 변화를 만든다. 이것은 실지의 장면이 보통 점광원에 의하여 조명되지 않으며 조명모형이 현실의 조명효과를 정밀하게 표현하는데 너무 간단하기때문이다.

도형처리프로그람들은 세기를 감쇠시키기 위하여 d의 선형 또는 2차함수값의 역수를 리용하는 방법으로 이 문제를 보상하였다. 실례로 일반적인 역2차감쇠함수는

$$f(d) = \frac{1}{a_0 + a_1 d + a_2 d^2} \tag{14-11}$$

과 같이 설정할수 있다. 사용자는 장면에 대한 여러가지 조명효과를 얻기 위하여 결수 a_0, a_1, a_2를 조종할수 있다. 상수항 a_0의 값은 d가 대단히 작을 때 $f(d)$가 너무 커지지 않도록 조정될수 있다. 또한 감쇠함수에서 결수들의 값과 장면에 대한 광학면파라메터들은 반사세기의 계산이 최대허용값을 넘지 않도록 조정될수 있다. 이것은 단일광원을 리용하여 장면을 조명할 때 세기값을 제한하는 효과적인 방법이다. 다중광원을 리용하여 조명할 때 앞에서 설명한 방법들은 세기범위를 제한하는데 더 효과적이다.

감쇠결수들의 주어 진 모임에 대하여 감쇠함수의 크기계산은

$$f(d) = \min\left(1, \frac{1}{a_0 + a_1 d + a_2 d^2}\right) \tag{14-12}$$

에 의해 1로 제한할수 있다. 이 함수를 리용하여 기본조명모형을

$$I = k_a I_a + \sum_{i=1}^{n} f(d_i) I_{li} \left[k_d (\mathbf{N}\cdot\mathbf{L}_i) + k_s (\mathbf{N}\cdot\mathbf{H}_i)^{n_s}\right] \tag{14-13}$$

과 같이 쓸수 있다. 여기서 d_i는 빛이 광원으로부터 움직인 거리이다.

색의 고찰

현실감 있는 장면들의 대부분의 도형처리현시들은 색갈이 있다. 그러나 이때까지 설명한 조명모형은 단색조명효과만을 고찰하였다. 색을 병합하기 위하여 세기식을 광원과 물체면의 색특성의 함수로 고찰하여야 한다.

RGB표현에서 장면의 매개 색은 붉은색, 풀색, 푸른색성분들에 의하여 표현된다. 그러면 광원의 세기와 면의 색갈의 RGB성분들을 지적하고 조명모형은 반사빛의 RGB성분들을 계산한다. 면의 색갈을 설정하는 한가지 방법은 반사률을 3원소벡토르로 지적하는것이다. 그러면 실례로 확산반사결수벡토르는 RGB성분(k_{dR}, k_{dG}, k_{dB})을 가진다. 물체가 푸른색면을 가지기를 원한다면 푸른색반사률성분 k_{dB}에 대하여 0～1사이 범위의 비령값을 선택하고 한편 붉은색 및 풀색반사률성분들은 령($k_{dR}=k_{dG}=0$)으로 설정한다. 입사빛의 임의의 비령붉은색 또는 풀색성분들은 흡수되며 푸른색성분만이 반사된다. 이 실례에 대한 세기계산은 단일식

$$I_B = k_{aB} I_{aB} + \sum_{i=1}^{n} f_i(d) I_{lBi} [k_{dB} (\mathbf{N}\cdot\mathbf{L}_i) + k_{sB} (\mathbf{N}\cdot\mathbf{H}_i)^{n_s}] \tag{14-14}$$

로 줄어 든다. 일반적으로 면들은 백색광원에 의하여 조명되며 면의 색갈을 반사빛이 세개의 모든 RGB성분들에 대하여 비령값을 가지도록 설정할수 있다. 매개 색성분에 대하여 계산된 세기준위는 RGB현시장치의 대응하는 전자총을 조정하는데 리용될수 있다.

자기의 본래 거울반사모형에서 퐁은 파라메터 k_s를 면의 색에 독립인 상수값으로 설정하였다. 이것은 입사빛과 같은 색(보통 흰색)의 거울반사를 만들며 그것은 면에 인위적인 감을 준다. 비소성재질에 대하여 거울반사의 색은 면의 특성들의 함수이며 그것은 입사빛의 색과도 다르고 확산반사의 색과도 다를수 있다. 이런 면에서의 거울효과를 식 14-14에서와 같이 거울반사결수를 색에 관계되도록 하여 근사화할수 있다. 그림 14-21은 무광택면에서의 색반사를 보여 주며 그림 14-22와 14-23은 금속면에서의 색반사를 보여 준다. 다중색광원으로 인한 물체면에서의 빛반사를 그림 14-24에 보여

주었다.

그림 14-21. 검은 나이론방석면에서의 빛반사(뜨개천무늬로 모형화하고 몬떼까를로광선추적방법을 리용하여 실감처리하였다.)

그림 14-22. 윤을 없앤 알루미니움면을 모의하기 위하여 설정된 반사파라메터를 가지는 주전자에서의 빛반사(몬떼까를로광선추적방법을 리용하여 실감처리하였다.)

그림 14-23. 광택황동면을 모의하기 위하여 설정된 반사파라메터들을 가지는 트롬본에서의 빛반사

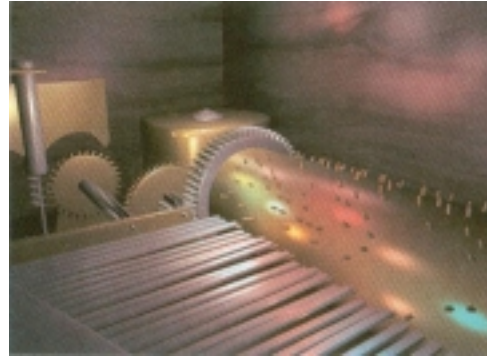



그림 14-24. 여러가지 색갈의 다중광원으로 인한 빛반사

면의 색을 설정하는 다른 방법은 매면에 대하여 확산 및 거울색벡토르들의 성분을 지적하고 한편 반사률을 단일상수값으로 보유하는것이다. 실례로 RGB색표현에 대하여 이 두개의 면색벡토르들의 성분은 (S_{dR}, S_{dG}, S_{dB}), (S_{sR}, S_{sG}, S_{sB})로 표시할수 있다. 그러면 반사빛의 푸른색성분은

$$I_B = k_a S_{dB} I_{aB} + \sum_{i=1}^{n} f_i(d) I_{lBi} [k_d S_{dB} (\mathbf{N} \cdot \mathbf{L}_i) + k_s S_{sB} (\mathbf{N} \cdot \mathbf{H}_i)^{n_s}] \qquad (14-15)$$

와 같이 계산된다. 이 방법은 약간 더 큰 적응성을 제공한다. 왜냐하면 면의 색파라메터들은 반사률값에 독립으로 설정할수 있기때문이다.

RGB이외의 다른 색표현들이 장면안의 색을 표현하는데 리용될수 있다. 그리고 색을 지적할 때 3개이상의 성분들을 가지는 색모형을 리용하는것이 때때로 편리하다. 색모형은 다음장에서 자세히 설명한다. 당분간 색의 임의의 성분을 그의 스펙트르파장 λ에 의하여 간단히 표현할수 있다. 그러면 세기계산은

$$I_\lambda = k_a S_{d\lambda} I_{a\lambda} + \sum_{i=1}^{n} f_i(d) I_{l\lambda i} [k_d S_{d\lambda} (\mathbf{N} \cdot \mathbf{L}_i) + k_s S_{s\lambda} (\mathbf{N} \cdot \mathbf{H}_i)^{n_s}] \qquad (14-16)$$

과 같이 표현할수 있다.

투명도

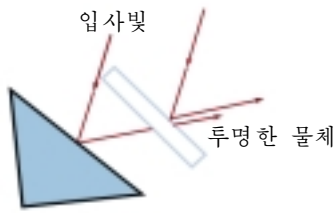



그림 14-25. 투명한 면에서의 빛 방출은 일반적으로 반사빛과 투과빛의 결합이다.

일반적으로 투명한 면은 반사빛과 투과빛을 다같이 만든다. 투과빛의 상대적인 비률은 면의 투명도에 관계되며 또한 광원이나 조명되는 면이 투명한 면의 뒤에 있는가에 관계된다. 그림 14-25는 투명한 물체의 면에서 나오는 빛의 조성을 보여 준다.

투명한 면을 모형화할 때 세기식은 면을 통과하는 빛을 고려하도록 수정하여야 한다. 대부분의 경우 투과빛은 그림 14-26에서와 같이 면의 뒤에 있는 반사물체로부터 발생된다. 이런 물체들로부터의 반사빛은 투명한 면을 통과하며 면의 전체 세기에 이바지한다.

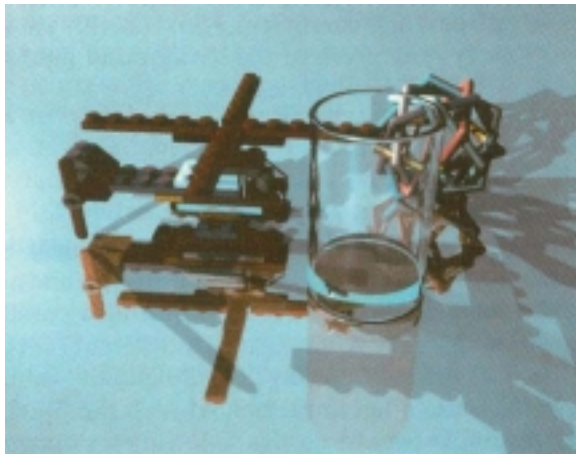



그림 14-26. 투명한 유리면의 광선추적된 보임상(유리의 뒤에 있는 물체로부터의 빛투과와 유리면으로부터의 빛반사를 다같이 보여 준다.)

투명한 물체의 면에서 확산 및 거울투과는 다같이 일어 날수 있다. 확산효과는 특히 서리가 내린 유리와 같은 투명한 면을 모형화할 때 중요하다. 이런 재질을 통과하는 빛은 배경물체의 희매해 진상이 얻어 지도록 산란된다. 확산굴절은 굴절빛의 세기를 감소시키고 굴절면의 매점에서의 세기를 유한구역에로 확산시켜 만들수 있다. 이 처리는 시간이 오래 걸리며 대부분의 조명모형들은 거울효과만을 쓴다.

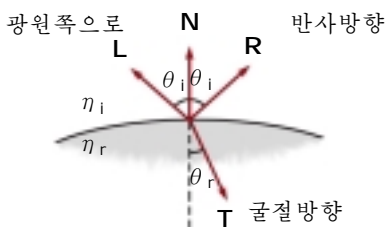



그림 14-27. 굴절률이 η_r인 면에 입사하는 광선에 대한 반사방향 **R**와 굴절방향 **T**

현실감 있는 투명효과는 빛의 굴절을 고찰하여 모형화한다. 빛이 투명한 면우에 입사할 때 일부는 반사되고 일부는 굴절된다(그림 14-27). 빛의 속도는 서로 다른 재질들에서 서로 다르기때문에 굴절빛의 경로는 입사빛과 다르다. 굴절각에 의하여 지적되는 굴절빛의 방향은 매개 재질의 굴절률과 입사빛의 방향의 함수이다. 재질의 굴절률은 재질에서의 빛속도에 대한 진공에서의 빛속도의 비로 정의된다. 굴절각 θ_r는 입사각 θ_i, 입사재질(보통 공기)의 굴절률 η_i, 굴절재질의 굴절률 η_r로부터 스넬의 법칙(Snell's law)에 따라 계산된다.

$$\sin\theta_r = \frac{\eta_i}{\eta_r}\sin\theta_i \qquad (14-17)$$

실제로 재질의 굴절률은 입사빛의 파장의 함수이다. 그러므로 광선의 서로 다른 색성분들은 서로 다른 각으로 굴절될것이다. 대부분의 응용들에서는 장면안에서 모형화되는 서로 다른 재질들의 평균

굴절률을 리용할수 있다. 공기의 굴절률은 약 1이고 크라운유리의 굴절률은 약 1.5이다. 식 14-17에서 이 값들을 리용하면 입사각이 30°일 때 굴절각은 약 19°이다. 그림 14-28은 유리를 통하여 굴절되는 광선에 대한 경로방향에서의 변화를 보여 준다. 전체적인 굴절효과는 입사빛을 평행으로 미는것이다. 식 14-17의 삼각함수계산에 시간이 걸리므로 굴절효과는 입사빛의 경로를 간단히 약간 미는 방법으로 모형화할수 있다.

스넬의 법칙과 그림 14-27로부터 굴절방향 θ_r에서의 단위투과벡토르 **T**를

$$\mathbf{T} = \left(\frac{\eta_i}{\eta_r}\cos\theta_i - \cos\theta_r\right)\mathbf{N} - \frac{\eta_i}{\eta_r}\mathbf{L} \tag{14-18}$$

과 같이 얻을수 있다. 여기서 **N**은 면의 단위법선벡토르이고 **L**은 광원방향에서의 단위벡토르이다. 투과벡토르 **T**는 굴절경로와 투명한 면뒤의 물체와의 사귐점을 찾는데 리용될수 있다. 굴절효과를 포함하면 장면을 대단히 현실감 있게 현시할수 있지만 굴절경로와 물체와의 사귐점의 결정은 상당한 계산을 요구한다. 대부분의 주사선화상공간방법들은 처리시간을 줄이는 근사방법으로 빛투과를 모형화한다. 광선추적알고리듬(14장 6절)에서 굴절을 다시 설명한다.

그림 14-28. 유리를 통한 빛의 굴절 (나오는 굴절광선은 입사빛의 경로(파선)에 평행인 경로를 따라 움직인다.)

투명한 물체를 모형화하는 더 간단한 절차는 경로밀기를 아주 무시하는것이다. 사실상 이 방법은 한 재질로부터 다른 재질로 넘어 갈 때 굴절률에서 변화가 없다는것을 가정하며 그리하여 굴절각은 항상 입사각과 같다. 이 방법은 세기계산의 속도를 높이며 얇은 다각형면에 대하여 합리적인 투명효과를 만들수 있다.

배경물체로부터 면을 통한 투과세기 I_{trans}는 투명결수 k_t를 리용하여 투명한 면에서의 반사세기 I_{refl}와 결합할수 있다(그림 14-29). 배경빛이 투과되는 량을 지적하기 위하여 파라메터 k_t에 0~1사이의 값을 할당한다. 그러면 면의 전체 세기는

$$I = (1-k_t)I_{\text{refl}} + k_t I_{\text{trans}} \tag{14-19}$$

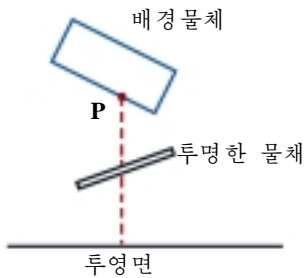
와 같이 계산된다. 항 $(1-k_t)$는 불투명결수이다.

배경물체
P
투명한 물채
투영면

그림 14-29. 배경물체의 점 **P**에서의 세기는 수직투영선(파선)을 따라 투명한 물체면에서의 반사세기와 결합될수 있다.

대단히 투명한 물체에 대하여 k_t에는 1에 가까운 값을 할당한다. 거의 불투명한 물체들은 배경물체로부터 대단히 적은 빛을 투과시키며 이런 재질들에 대하여 k_t에는 0에 가까운 값(1에 가까운 불투명도)을 설정할수 있다. k_t는 면우의 점의 함수로도 될수 있다. 그러므로 물체의 서로 다른 부분들은 k_t에 할당된 값에 따라 많거나 적은 배경세기를 투과시킬수 있다.

투명효과는 자주 수정된 깊이완충기(z완충기)알고리듬에 의하여 실현된다. 이렇게 하는 간단한 방법은 먼저 보이는 불투명면에 대한 깊이를 결정하기 위하여 불투명물체들을 처리하는것이다. 다음에 투명한 물체의 깊이위치는 깊이완충기에 앞서 기억된 값과 비교된다. 임의의 투명한 면이 보이면 그의 반사세기가 계산되고 프레임완충기에 앞서 기억된 불투명면의 세기와 결합된다. 이 방법은 투명한 면의 깊이와 기타 파라메터들에 대한 추가적인 기억기를 리용하여 보다 정밀하게 현시할수 있도록 수정할수 있다. 이것은 투명한 면의 깊이값이 불투명한 면의 깊이값뿐아니라 자기자체와도 서로서로 비교되게 한다. 보이는 투명한 면은 다음에 그의 면세기를 뒤에 있는 보이는 불투명면들과 결합하여 실감처리된다.

투명성의 정밀한 현시와 경계허상제거는 A완충기알고리듬에 의하여 얻을수 있다. 매개 화소위치에서 모든 겹침면들에 대한 면쪼각들은 보관되고 깊이순서로 정렬된다. 다음에 깊이에서 겹치는 투명 및 불투명면쪼각들에 대한 세기는 13장에서 설명한바와 같이 화소에 대한 마지막평균세기를 만들기 위하여 적당한 보임성순서로 결합된다.

깊이정렬보임성알고리듬은 먼저 면들을 깊이순서로 정렬하고 다음에 어느 보이는 면이 투명한가를 결정하여 투명성을 처리하도록 수정될수 있다. 보이는 투명한 면을 발견하면 그의 반사된 면세기는 매개 투영면점에서의 화소세기를 얻기 위하여 뒤에 있는 물체들의 면세기와 결합된다.

그림자

보이지 않는 면방법들은 광원이 그림자를 만드는 구역을 찾는데 리용될수 있다. 보기위치에 광원이 있는 보이지 않는 면방법을 적용하여 어느 면부분들이 광원으로부터 볼수 없는가를 결정할수 있다. 이것은 그림자구역들이다. 모든 광원들에 대한 그림자구역들을 결정하였으면 그림자는 면무늬로 취급되며 무늬배렬에 기억된다. 그림 14-30에서는 탁우의 두 물체와 먼 거리의 광원에 대하여 음영무늬의 발생을 보여 주었다. 이 그림의 모든 그림자구역들은 광원위치로부터 보이지 않는 면들이다. 그림 14-26의 장면은 다중광원에 의하여 만들어 지는 그림자효과를 보여 준다.

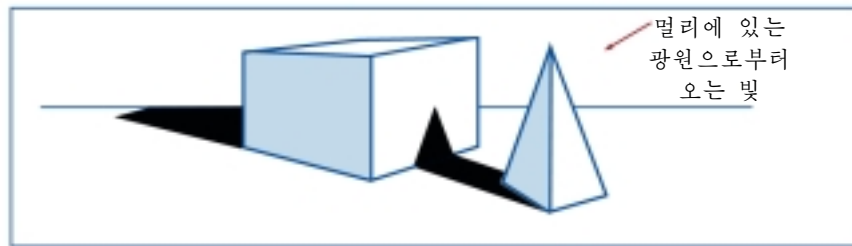



그림 14-30. 그림자구역에 의하여 모형화 되는 물체

보이지 않는 면방법에 의하여 발생되는 그림자무늬는 광원의 위치가 변화되지 않는 한 임의로 선택되는 보기위치에 대하여 유효하다. 보기위치로부터 보이는 면들은 조명모형에 따라 명암처리되며 그것들은 결문양과 결합될수 있다. 그림자구역은 주변빛세기만으로 현시할수 있으며 또는 주변빛을 면의 지적된 결문양과 결합할수 있다.

3절. 빛세기의 현시

조명모형에 의하여 계산된 세기값은 사용하는 개개의 도형처리체계에서 허용가능한 세기준위값으로 변환되여야 한다. 일부 체계들은 여러가지 세기준위를 현시할수 있으며 한편 다른것들은 매 화소에 대하여 두개의 준위(on 또는 off)만 가능하다. 첫번째 경우에 조명모형으로부터 계산된 세기는 프레임완충기에서 사용가능한 세기준위로 변환된다. 2값체계에서 세기는 다음절에서 설명하는바와 같이 중간명암무늬로 변환된다.

세기준위의 할당

먼저 현시장치에서 회색계조값의 분포가 등세기간격의 감각에 대응하도록 회색계조값들이 어떻게 0~1사이의 범위에 분포될수 있는가를 고찰한다. 상대빛세기는 상대음세기를 감각하는것과 같은 방법 즉 로그척도로 감각된다. 이것은 두 세기의 비가 다른 두 세기의 비와 같으면 매개 세기쌍에서의 차이를 같다고 감각한다는것을 의미한다. 실례로 세기 0.20과 0.22사이의 차이는 0.80과 0.88사이

의 차이와 같다고 감각한다. 그러므로 현시장치에서 n+1개의 련속한 세기준위를 등감각밝기로 현시하자면 련속하는 세기들의 비가 일정하도록 세기준위들의 간격을 설정하여야 한다.

$$\frac{I_1}{I_0}=\frac{I_2}{I_1}=\cdots=\frac{I_n}{I_{n-1}}=r \qquad (14-20)$$

여기서 현시장치에 현시될수 있는 제일 낮은 준위를 I_0으로 표시하고 제일 높은 준위를 I_n으로 표시한다. 임의의 중간세기는 I_0에 의하여

$$I_k=r^kI_0 \qquad (14-21)$$

과 같이 표현할수 있다. 개개의 체계에 대하여 I_0과 n의 값이 주어 지면 앞의 식에 k=n을 대입하여 r의 값을 계산할수 있다. I_n=1이기때문에

$$r=\left(\frac{1}{I_0}\right)^{1/n} \qquad (14-22)$$

이다. 그리하여 식 14-21에서 I_k에 대한 계산은

$$I_k=I_0^{(n-k)/n} \qquad (14-23)$$

과 같이 쓸수 있다. 실례로 n=3인 체계에서 I_0=1/8이면 r=2이며 4개의 세기값은 1/8, 1/4, 1/2, 1이다.

제일 낮은 세기값 I_0은 현시장치의 특성에 관계되며 일반적으로 0.005～약 0.025범위에 있다. 2장에서 본바와 같이 현시장치에 현시되는 《검은》구역은 화면형광체에서 반사되는 빛때문에 항상 0이상의 어떤 세기값을 가질것이다. 화소당 8bit(n=255)이고 I_0=0.01인 흑백현시장치에서 련속하는 세기들의 비는 대략 r=1.0182이다. 이 체계에서 256개의 세기에 대한 근사값은 0.0100, 0.0102, 0.0104, 0.0106, 0.0107, 0.0109, ..., 0.9821, 1.0000이다.

색체계에서 색모형의 매개 성분에 대하여 세기준위를 설정한다. 실례로 RGB모형을 리용할 때 식 14-21에서와 같이 준위가 k인 세기의 푸른색성분은 얻을수 있는 제일 낮은 푸른색값에 관련시킬수 있다.

$$I_{Bk}=r_B^kI_{B0} \qquad (14-24)$$

여기서

$$r_B=\left(\frac{1}{I_{B0}}\right)^{1/n} \qquad (14-25)$$

이며 n은 세기준위들의 개수이다. 류사한 식은 다른 색성분들에 대해서도 성립한다.

감마보정과 영상검색표

계산된 세기의 현시와 관련되는 다른 문제는 현시장치의 비선형성이다. 조명모형은 선형적인 세기들을 만든다. 조명모형으로부터 얻어 지는 RGB색(0.25, 0.25, 0.25)은 색(0.5, 0.5, 0.5)의 1/2세기를 표현한다. 보통 이렇게 계산된 세기들은 화상파일에서 매개 세기의 RGB성분들에 대하여 한 바이트씩 옹근수값으로 기억된다. 이 세기파일은 역시 선형이며 따라서 값(64, 64, 64)를 가지는 화소는 값(128, 128, 128)을 가지는 화소의 1/2세기를 가진다. 그러나 현시장치는 비선형장치이다. 전자총의 전압을 선형화소값에 비례하도록 설정하면 세기는 그림 14-31에 보여 준 현시장치응답곡선에 따라 밀려 질것이다.

현시장치의 비선형성을 보정하기 위하여 도형처리체계들은 선형화소값을 조정하는 영상검색표를 리용한다. 현시장치의 응답곡선은 지수함수

$$I = aV^{\gamma} \tag{14-26}$$

으로 표현된다. 파라메터 I는 현시되는 세기, 파라메터 V는 입구전압이다. 파라메터 a와 γ의 값은 도형처리체계에서 사용하는 현시장치의 특성에 관계된다. 그러므로 특별한 세기값 I를 현시하려 할 때 이 세기를 만드는 정확한 전압값은

$$V = \left(\frac{I}{a}\right)^{1/\gamma} \tag{14-27}$$

이다. 이 계산을 세기의 **감마보정**(gamma correction)이라고 한다. 현시장치의 감마값은 일반적으로 2.0～3.0사이에 있다. 민족텔레비존체계위원회(National Television System Committee, NTSC)신호표준은 $\gamma=2.2$이다. 그림 14-32에서는 NTSC감마값을 리용하는 감마보정곡선을 보여 주었다. 식 14-27은 화상파일안의 옹근수화소값을 전자총의 전압을 조종하는 값으로 변환하는 영상검색표를 설정하는데 리용된다.

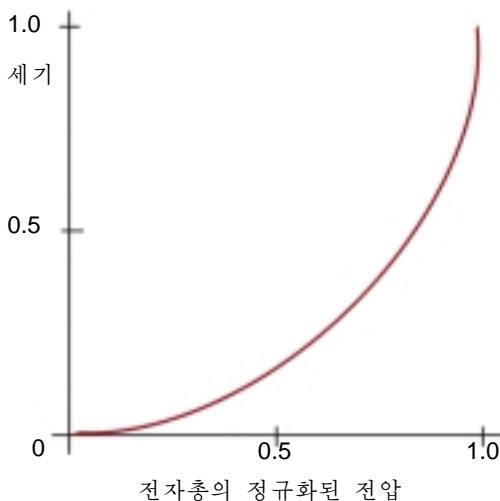



그림 14-31. 현시되는 화면세기를 전자총의 정규화된 전압의 함수로 보여 주는 일반적인 현시장치응답곡선

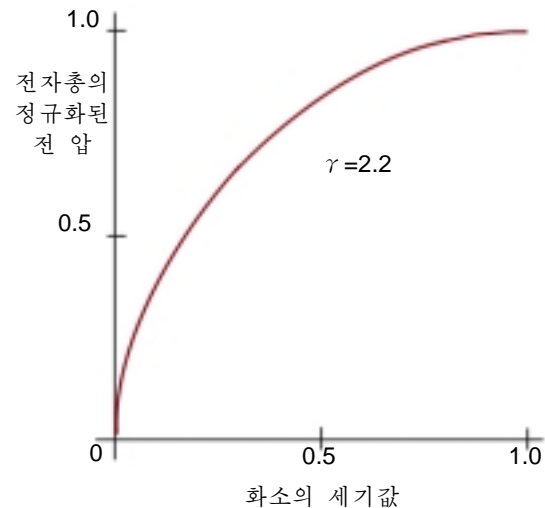



그림 14-32. $\gamma=2.2$인 감마보정을 리용하여 화소세기를 전자총의 전압으로 넘기는 영상검색보정곡선(화소세기와 현시장치의 전압값은 다같이 0～1 구간에서 정규화된다.)

두 변환을 다같이 포함하는 검색표를 만들기 위하여 감마보정과 로그세기넘기기를 결합할수 있다. I가 조명모형으로부터 계산된 세기값이면 먼저 식 14-20 또는 14-23에 의하여 만들어 지는 값들의 표로부터 제일 가까운 세기 I_k를 찾아 낸다. 한편 이 세기값의 준위번호는

$$k = \mathrm{round}\left(\log_r \frac{I}{I_0}\right) \tag{14-28}$$

에 의해 결정하고 식 14-23을 리용하여 이 준위의 세기값을 계산할수도 있다. 세기값 I_k를 가지면 전자총의 전압은

$$V_k = \left(\frac{I_k}{a}\right)^{1/\gamma} \tag{14-29}$$

로 계산할수 있다. 값 V_k는 다음에 검색표에 넣어 질수 있으며 k의 값은 프레임완충기의 화소위치에 기억된다. 특별한 체계가 검색표를 가지지 않으면 V_k의 계산된 값은 프레임완충기에 직접 기억될수

있다. 때때로 로그세기넘기기와 식 14-29를 리용하는 V_k의 계산이 결합된 변환을 감마보정이라고도 한다.

현시장치의 영상증폭기가 선형입력화소값을 전자총의 전압으로 변환하도록 설계되면 두개의 세기변환처리를 결합할수 없다. 이 경우에 감마보정은 하드웨어안에 만들어 지며 로그값 I_k는 미리 계산되고 프레임완충기(또는 색표)에 기억되여야 한다.

련속색조화상의 현시

고급콤퓨터도형처리체계들은 일반적으로 매 색성분에 대하여 256개의 세기준위를 제공하지만 더 적은 준위를 가지는 응용들에서도 적당한 현시를 얻을수 있다. 4계조체계는 련속색조화상에 대한 최소명암능력을 제공하며 한편 사진같은 화상들은 화소당 32～256개의 세기준위능력이 있는 체계들에서 만들어 질수 있다.

그림 14-33에서는 여러가지 세기준위에 의하여 현시되는 련속색조사진을 보여 주었다. 적은 개수의 세기준위를 리용하여 련속색조화상을 다시 만들 때 서로 다른 세기구역들사이에 경계선(륜곽선)이 나타난다. 2값재생성에서 사진의 특징은 겨우 식별할수 있다. 4개의 세기준위를 리용할 때 본래의 명암무늬를 식별할수 있지만 륜곽선효과는 뚜렷하다. 8개의 세기준위를 리용할 때 륜곽선효과는 여전히 뚜렷하지만 본래의 명암을 더 좋게 표현하기 시작한다. 16 또는 그이상의 세기준위들에서 륜곽선효과는 축소되며 표현은 본래의것에 대단히 가깝다. 32개이상의 세기준위를 리용하는 련속색조화상의 재생성은 본래의것과 거의 차이가 없다.

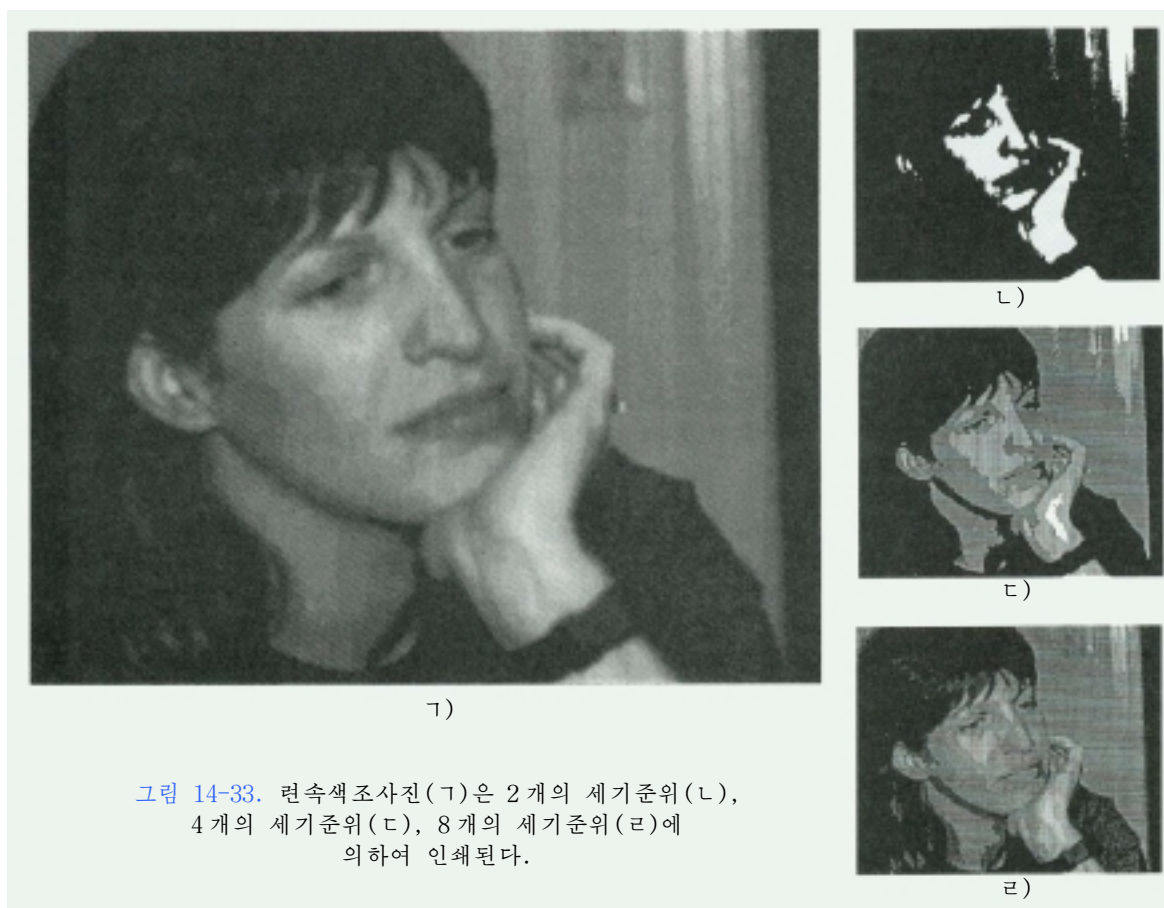



그림 14-33. 련속색조사진(ㄱ)은 2개의 세기준위(ㄴ), 4개의 세기준위(ㄷ), 8개의 세기준위(ㄹ)에 의하여 인쇄된다.

4절. 중간명암무늬와 한정색표시기술

출력장치가 제한된 세기범위를 가질 때 한개의 세기값을 현시하는데 여러개의 화소위치들을 리용하여 겉으로 보기에 사용가능한 세기들의 개수를 증가시킬수 있다. 여러개의 화소위치들로 이루어지는 작은 구역을 볼 때 사람의 눈은 미세한 세부를 전체적인 밝기에 통합 또는 평균하는 경향이 있다. 2값현시장치 및 인쇄기들은 많은 세기값들을 가지는 그림을 만드는데 이 시각효과의 우점을 리용한다.

신문, 잡지, 책들의 출판에서 련속색조사진은 **중간명암처리**(halftoning)라고 하는 인쇄처리에 의하여 재생성되며 재생성된 그림은 중간명암(halftone)이라고 한다. 흑백사진에 대하여 매개 세기구역은 흰 배경우에 검은 원들로 재생성된다. 매개 원의 직경은 그 세기구역에서 요구되는 어두움에 비례한다. 보다 어두운 구역은 보다 큰 원에 의하여 인쇄되며 보다 밝은 구역은 보다 작은 원(보다 흰 구역)에 의하여 인쇄된다. 그림 14-34에서는 중간명암처리방법에 의하여 재생성된 흑백사진의 확대된 부분을 보여 주었다. 그림 14-35에 보여 준바와 같이 색중간명암은 여러가지 크기와 색갈의 점들을 리용하여 인쇄된다. 책과 잡지의 중간명암은 대략 센치메터당 60∼80개의 서로 다른 직경의 원들을 리용하여 고급종이에 인쇄된다. 신문은 더 낮은 급의 종이와 더 낮은 분해능(센치메터당 약 25∼30점)을 리용한다.

그림 14-34. 중간명암처리방법에 의하여 재생성된 사진의 확대된 부분(색조들이 여러가지 크기의 점들에 의하여 어떻게 표현되는가를 보여 준다.)

그림 14-35. 색중간명암의 점무늬(색중간명암 ㄱ에 있는 시계의 웃절반은 ㄴ에서 10배, ㄷ에서 50배로 확대된다.)

중간명암근사

콤퓨터도형처리에서 중간명암재생성은 중간명암무늬(halftone pattern) 또는 화소무늬(pixel pattern)라고 부르는 직4각형화소구역들을 리용하여 근사된다. 이 방법에 의하여 현시할수 있는 세기준위들의 개수는 직4각형격자안에 얼마나 많은 화소들이 있으며 체계가 얼마나 많은 준위들을 현시할수 있는가에 관계된다. 2값체계에서 매개 격자의 $n \times n$화소들은 n^2+1개의 세기준위들을 표현할수 있다. 그림 14-36에서는 2값체계에서 리용할수 있는 5개의 세기준위를 표현하기 위하여 화소무늬들을 설정하는 한가지 방법을 보여 주었다. 무늬 0에서 모든 화소들은 off, 무늬 1에서 한개의 화소가 on, 무늬 4에서 4개의 모든 화소들이 on이다. 장면안의 세기값 I는 그림에 보여 준 매 격자의 아래에 기입된 범위에 따라 개개의 무늬에로 넘겨 진다. 무늬 0은 $0 \leqq I < 0.2$, 무늬 1은 $0.2 \leqq I < 0.4$, 무늬 4는 $0.8 \leqq I \leqq 1.0$에 리용된다.

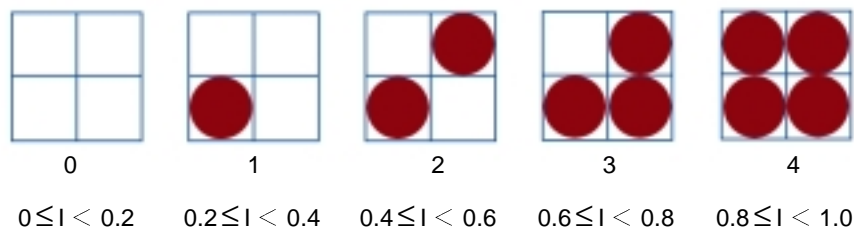



그림 14-36. 2값체계에서 5개의 세기준위를 현시하는데 리용되는 2×2화소격자(매개 격자에 넘겨 지는 세기값은 매개 화소무늬의 아래에 기입한다.)

2값체계에서 3×3화소격자에 의하여 10개의 세기준위를 현시할수 있다. 이 준위들에 대하여 10개의 화소무늬를 설정하는 한가지 방법을 그림 14-37에 보여 주었다. 매개 준위에서 화소위치들은 중간명암재생성에 리용되는 증가하는 크기의 원을 근사하도록 선택된다. 즉 on화소위치들은 낮은 세기준위들에 대하여 격자중심의 가까이에 있으며 세기준위가 증가할 때 바깥쪽으로 확대된다.

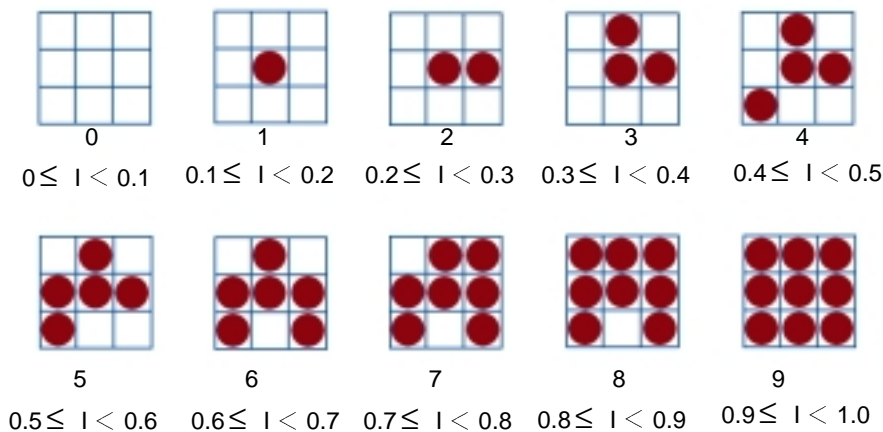



그림 14-37. 3×3화소격자는 2값체계에서 10개의 세기준위를 현시하는데 리용될수 있다.(매개 격자에 넘겨 지는 세기값은 매개 화소무늬의 아래에 기입한다.)

임의의 크기의 화소격자에서는 화소의 위치번호들의 마스크에 의해 여러가지 가능한 세기준위들의 화소무늬를 표현할수 있다. 실례로 다음의 마스크는 그림 14-37에 보여 준 0이상의 세기준위들에 대하여 9개의 3×3화소무늬를 만드는데 리용할수 있다.

$$\begin{bmatrix} 8 & 3 & 7 \\ 5 & 1 & 2 \\ 4 & 9 & 6 \end{bmatrix} \qquad (14-30)$$

번호가 k인 세기준위를 현시하기 위하여 위치번호가 k보다 작거나 같은 매개 화소를 on으로 한다.

$n \times n$화소무늬를 리용할 때 현시될수 있는 세기들의 수는 증가하지만 현시되는 그림의 분해능은 매 x, y축을 따라 $1/n$배로 줄어 든다. 실례로 512×512 화면구역은 2×2화소무늬에 의하여 256×256개의 세기점을 포함하는 구역으로 줄어 든다. 3×3화소무늬에 의하여 512×512구역은 매변을 따라 128개의 세기위치로 줄어 든다.

화소격자의 다른 문제는 격자의 크기가 증가할 때 부분격자의 무늬가 보이는것이다. 세기준위들을 이지러짐이 없이 현시하는데 리용할수 있는 격자의 크기는 현시되는 화소의 크기에 관계된다. 분해능이 낮은 체계(센치메터당 적은 화소)들에서는 세기준위의 개수가 작아야 한다. 한편 고급한 현시는 적어도 64개의 세기준위를 요구한다. 이것은 8×8화소격자가 요구된다는것을 의미한다. 그리고 책 및 잡지들에서의 중간명암과 같은 분해능을 만들자면 센치메터당 60점을 현시하여야 한다. 따라서 센치메터당 60×8=480점을 현시할수 있어야 한다. 일부 장치들 실례로 고급필림기록기는 이 분해능을 현시할수 있다.

중간명암근사에 리용하는 화소무늬는 본래의 장면에 없는 륜곽선 및 다른 시각효과들을 최소화하도록 만들어야 한다. 륜곽선은 매개 련속하는 화소무늬를 앞의 무늬로부터 끌어 내여 최소화할수 있다. 즉 준위 k의 무늬는 준위 k-1의 화소무늬에 on위치를 하나 추가하여 만든다. 그러므로 한 화소위치가 하나의 격자준위에 대하여 on이면 그것은 더 높은 모든 준위들에 대하여 on이다(그림 14-36과 14-37). 다른 시각효과들은 대칭무늬를 피하면 최소화할수 있다. 실례로 3×3화소격자에서 세번째 세기준위는 그림 14-38 ㄴ의 임의의 대칭배렬보다 그림 14-38 ㄱ의 무늬를 리용하면 더 잘 표현된다. 이 그림의 대칭무늬들은 세기준위 3에 의한 큰 명암구역에서 수직, 수평, 대각줄무늬를 만들것이다. 필림기록기와 일부 인쇄기와 같은 장치들의 경복사출력에서 고립된 화소들은 효과적으로 재생성되지 않는다. 따라서 그림 14-39에서와 같이 단일 on화소 또는 고립된 on화소들을 가지는 격자무늬는 피해야 한다.

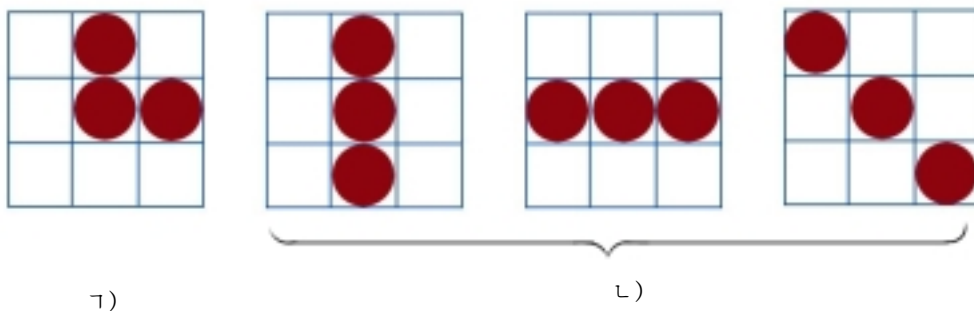



그림 14-38. 3×3화소격자의 세번째 세기준위의 표현에서
무늬 ㄱ는 ㄴ의 무늬들보다 더 좋다.

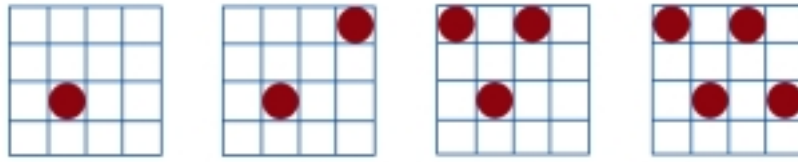



그림 14-39. 일부 경복사장치들에서 효과적으로 재생성될수 없는 고립된 화소들을 가지는 중간명암화소무늬

중간명암근사는 화소당 2개이상의 세기를 현시할수 있는 체계들에서 세기선택항의 수를 증가시키는데 리용할수 있다. 실례로 화소당 4개의 세기준위를 현시할수 있는 체계에서 2×2화소격자는 사용가능한 세기준위를 4로부터 13에로 확장하는데 리용할수 있다. 그림 14-36에서 0우의 4개의 화소무늬는 매개 화소위치가 0우의 3개의 세기값을 현시할수 있기때문에 여러가지 준위들을 표현한다. 그림 14-40에서는 13개의 서로 다른 준위들을 얻기 위하여 화소세기들을 할당하는 한가지 방법을 보여주었다. 개별적인 화소에 대한 세기준위는 0~3으로 표식되며 체계에 대한 전체적인 준위는 0~12로 표식된다.

그림 14-40. 4계조체계에서 2×2화소격자를 리용하는 중간명암근사에 의해 얻어 지는 0~12까지의 세기준위들

류사하게 색체계에서 현시할수 있는 세기들의 수를 증가시키는데 화소무늬를 리용할수 있다. 실례로 화소당 3bit의 RGB체계를 가진다고 하자. 이것은 현시장치에서 색전자총당 1bit를 주며(흑 및 백) 8개의 색을 제공한다. 2×2화소무늬를 리용하면 그림 14-41에 보여 준바와 같이 개개의 색값을 표현하는데 리용할수 있는 12개의 형광체점을 가진다. 3개의 매 RGB색은 무늬에서 4개의 형광체점을 가지며 그것은 색당 5개의 가능한 설정을 할수 있게 한다. 이것은 총 125개의 서로 다른 색조합을 준다.

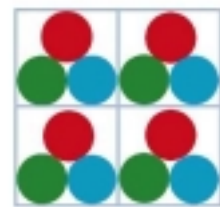



그림 14-41. RGB 2×2 화소무늬

한정색표시기술

용어 **한정색표시**(dithering)는 문맥들에서 여러가지로 리용된다. 처음에 그것은 분해능을 줄임이 없이 중간명암을 근사하는 기술을 말한다. 그러나 이 용어는 화소격자를 리용하는 중간명암근사방법에도 적용되며 때때로 색중간명암근사만을 말하는데도 리용된다.

륜곽선을 없애기 위하여 화소세기에 추가하는 우연값들을 데이자잡음이라고 한다. 우연값들을 분포시키기 위하여 여러가지 알고리듬들을 리용한다. 그 효과는 전체 그림에 잡음을 추가하는것이며 그것은 세기경계들을 부드럽게 하는 경향이 있다.

순서데이자방법(ordered-dither method)은 장면안의 점들을 현시화소에로 1대 1로 넘기면서 세기를 변화시킨다. n^2개의 세기준위를 얻기 위하여 $n \times n$데이자행렬 D_n을 설정하며 그의 원소들은 $0 \sim n^2-1$사이의 서로 다른 정의 옹근수이다. 실례로

$$D_2 = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \tag{14 – 31}$$

에 의하여 4개의 세기준위를 만들수 있으며

$$D_3 = \begin{bmatrix} 7 & 2 & 6 \\ 4 & 0 & 1 \\ 3 & 8 & 5 \end{bmatrix} \tag{14 – 32}$$

에 의하여 9개의 세기준위를 만들수 있다. D_2과 D_3의 행렬에서 원소들의 순서는 각각 2×2와 3×3화소격자를 설정하는 화소마스크에서와 같다. 2값체계일 때 현시세기값들은 입력세기와 행렬의 원소들을 비교하여 결정한다. 매개 입력세기는 먼저 $0 \le I \le n^2$범위에로 비례변환된다. 세기 I가 화면위치 (x, y)에 적용된다면 데이자행렬에 대한 행과 렬번호를

$$i = (x \bmod n) + 1, \qquad j = (y \bmod n) + 1 \tag{14–33}$$

과 같이 계산한다. $I > D_n(i,j)$이면 위치 (x, y)의 화소를 on으로 하고 그렇지 않으면 그 화소는 off로 한다.

데이자행렬의 원소들은 화소격자에서 설명한것과 같은 방법으로 할당한다. 즉 현시되는 장면에서 추가적인 시각효과를 최소화하려고 한다. 순서데이자는 그의 행렬의 원소값들을 격자마스크에 대응시킬 때 격자의 화소무늬가 만드는것과 동등한 상수세기구역을 만든다. 화소격자현시로부터의 변화는 세기준위들의 경계에서 일어 난다.

일반적으로 세기준위들의 개수는 2의 배수로 취한다. 더 큰 데이자행렬은 더 작은 데이자행렬로부터 재귀관계

$$D_n = \begin{bmatrix} 4D_{n/2} + D_2(1,1)U_{n/2} & 4D_{n/2} + D_2(1,2)U_{n/2} \\ 4D_{n/2} + D_2(2,1)U_{n/2} & 4D_{n/2} + D_2(2,2)U_{n/2} \end{bmatrix} \tag{14–34}$$

에 의하여 얻는다. $n \ge 4$라고 가정한다. 파라메터 $U_{n/2}$은 《1행렬》(모든 원소들이 1이다.)이다. 실례로 D_2가 식 14-31에서와 같이 지적되면 재귀관계 14-34는

$$D_4 = \begin{bmatrix} 15 & 7 & 13 & 5 \\ 3 & 11 & 1 & 9 \\ 12 & 4 & 14 & 6 \\ 0 & 8 & 2 & 10 \end{bmatrix} \tag{14–35}$$

를 낳는다.

$m \times n$개의 점들을 가지는 그림을 $m \times n$개의 화소를 가지는 현시구역에로 넘기는 다른 방법은 오차확산(error diffusion)이다. 여기서는 주어 진 위치에서의 입력세기값과 현시되는 화소의 세기준위사이의 오차를 현재 화소위치의 오른쪽과 아래의 화소위치에로 분산 또는 확산시킨다. 사진을 주사하여 얻어 진 세기값들의 행렬 **M**에서 시작하여 화면구역에 대한 화소의 세기값들의 배렬 **I**를 만들려고 한다. 먼저 왼쪽에서 오른쪽으로 우에서 아래로 **M**의 행들을 가로질러 주사하고 **M**의 매 원소에 대하여

제일 가까운 사용가능한 화소세기준위를 결정한다. 다음에 매개 화소위치에서 행렬 **M**에 기억된 값과 현시되는 세기준위사이의 오차를 다음의 간단화된 알고리듬을 리용하여 **M**의 린접원소들에 확산시킨다.

```
for(i=0; i<m; i++)
  for(j=0; j<n; j++){
          /*Determine the available intensity level I_k */
          /*that is closest to the value M_i,j. */
          I_i,j := I_k;
          err := M_i,j - I_i,j;
          M_i,j+1 := M_i,j+1 + α·err;
          M_i+1,j-1 := M_i+1,j-1 + β·err;
          M_i+1,j := M_i+1,j + γ·err;
          M_i+1,j+1 := M_i+1,j+1 + δ·err;
  }
```

행렬 **I**의 원소들에 세기준위값이 할당되였으면 그 행렬을 인쇄기나 현시장치의 일부 구역에 넘긴다. 물론 오차는 제일 마지막렬($j=n$)을 지나서 또는 제일 마지막행($i=m$)의 아래로 확산시킬수 없다. 2값체계에 대하여 사용가능한 세기준위는 0과 1이다. 오차를 확산시키는 파라메터들은 다음의 관계

$$\alpha+\beta+\gamma+\delta\leq 1 \qquad (14-36)$$

을 만족시키도록 선택할수 있다.

상당히 좋은 결과를 만드는 오차확산파라메터들에 대한 한가지 선택은 (α, β, γ, δ)=(7/ 16, 3/ 16, 5/ 16, 1/ 16)이다. 그림 14-42에서는 이 파라메터값들을 리용하는 오차확산을 보여 주었다. 오차확산은 때때로 그림의 일정한 부분 특히 이마에 머리칼이 난 언저리와 코의 륜곽선과 같은 얼굴특징들을 반복하여 그림에서 2중상을 만든다. 2중상은 오차확산파라메터들에 대하여 합이 1보다 작은 값들을 선택하고 오차확산후 행렬의 값들을 다시 비례변환하여 줄일수 있다. 다시 비례변환하는 한가지 방법은 **M**의 모든 원소들에 0.8을 곱하고 다음에 0.1을 더하는것이다. 그림의 질을 개선하는 다른 방법은 행렬의 행을 주사할 때 오른쪽왼쪽과 왼쪽오른쪽을 엇바꾸는것이다.

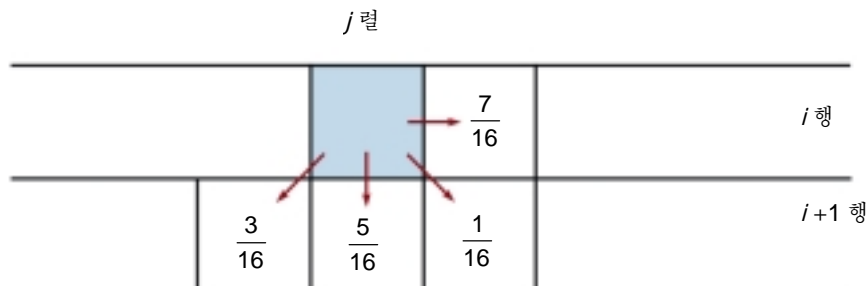



그림 14-42. 오차확산방법을 리용할 때 린접화소위치들에로 확산될수 있는 세기오차의 비률

오차확산방법의 변종은 점확산이다. 이 방법에서 세기값들의 $m \times n$배렬은 그림 14-43에 보여 준 바와 같이 0~63까지 번호 매겨 진 64개의 부분배렬로 나뉘여 진다. 행렬의 값과 현시되는 세기사이의 오차는 더 큰 부분배렬번호를 가지는 린접한 행렬원소들에만 확산된다. 64개의 부분배렬번호들의 분포는 더 낮은 부분배렬번호를 가지는 원소들에 의하여 완전히 둘러 싸이는 원소들의 개수를 최소화하는데 기초한다. 왜냐하면 이것은 둘러 싸인 원소들의 모든 오차를 그 한 위치에로 돌리는 경향이 있기때문이다.

34	48	40	32	29	15	23	31
42	58	56	53	21	5	7	10
50	62	61	45	13	1	2	18
38	46	54	37	25	17	9	26
28	14	22	30	35	49	41	33
20	4	6	11	43	59	57	52
12	0	3	19	51	63	60	44
24	16	8	27	39	47	55	36

그림 14-43. 세기배렬을 0~63까지 번호 붙인 64개의 점확산 부분배렬로 나누는 한가지 가능한 분포계획

5절. 다각형실감처리방법

이 절에서는 다각형면에 의하여 형성되는 표준적인 도형처리물체들의 실감처리에서 조명모형의 적용을 고찰한다. 물체들은 보통 곡면물체의 다각형그물근사이지만 그것들은 또한 곡면근사가 아닌 다면체일수 있다. 일반적으로 주사선알고리듬들은 다각형의 면을 실감처리하기 위하여 두가지 방법중 한가지로 조명모형을 적용한다. 매개 다각형은 단일세기로 실감처리될수 있으며 또는 면의 매점에서 보간계획을 리용하여 세기를 얻을수 있다.

상수세기명암처리

다각형면들을 가지는 물체를 실감처리하는 빠르면서도 간단한 방법은 **상수세기명암처리** (constant-intensity shading)이며 그것은 단색명암처리(flat shading)라고도 한다. 이 방법에서는 매 다각형에 대하여 단일세기가 계산된다. 다각형면우의 모든 점들은 동일한 세기값으로 현시된다. 단색명암처리는 그림 14-47에서와 같이 곡면의 일반적인 모양을 빨리 현시하는데 쓸모 있다.

일반적으로 다각형의 단색명암처리는 다음의 모든 가정들이 유효하면 물체를 정밀하게 실감처리한다.

- 물체는 다면체이며 곡면을 가지는 물체의 근사가 아니다.
- 물체를 조명하는 모든 광원들은 $\mathbf{N} \cdot \mathbf{L}$과 감쇠함수가 면우에서 일정하도록 면으로부터 충분히 멀리에 있다.
- 보기위치는 $\mathbf{V} \cdot \mathbf{R}$가 면우에서 일정하도록 면으로부터 충분히 멀다.

지어 이 모든 조건들이 참이 아니라도 여전히 면조명효과를 작은 다각형쪼각들을 리용하여 단색명암처리로 합리적으로 근사할수 있으며 매개 쪼각 즉 다각형의 중심에서의 세기를 계산할수 있다.

그로우명암처리

그로우에 의하여 개발되고 일반적으로 **그로우명암처리**(Gouraud shading)라고 하는 이 세기보간계획은 면을 가로질러서 세기값들을 선형보간하여 다각형의 면을 실감처리한다. 매개 다각형에 대한 세기값은 공통변을 따라 이웃한 다각형의 값과 정합되며 이리하여 단색명암처리에서 일어 날수 있는 세기의 불련속성을 없앤다.

매개 다각형면은 그로우명암처리에서 다음의 계산들을 수행하여 실감처리된다.

- 다각형의 매 정점에서 평균단위법선벡토르를 결정한다.
- 정점의 세기를 계산하기 위하여 조명모형을 매개 정점에 적용한다.
- 다각형의 면우에서 정점의 세기들을 선형보간한다.

그림 14-44에서 보여 주는바와 같이 매개 다각형정점에서 그 정점을 공유하는 모든 다각형면들의 법선들을 평균하여 법선벡토르를 얻는다. 이리하여 임의의 정점위치 **V**에서

$$\mathbf{N}_V = \frac{\sum_{k=1}^{n} \mathbf{N}_k}{\left|\sum_{k=1}^{n} \mathbf{N}_k\right|} \qquad (14-37)$$

에 의해 정점의 단위법선을 얻는다. 정점의 법선을 알면 조명모형으로부터 정점에서의 세기를 결정할 수 있다.

그림 14-45에서는 다각형의 변을 따라 세기를 보간하는 다음번 걸음을 보여 주었다. 매개 주사선에 대하여 주사선과 다각형의 변과의 사귐점에서의 세기는 변의 끝점들에서의 세기로부터 선형보간된다. 그림 14-45의 실례에서 위치 1과 2에서 끝점을 가지는 다각형의 변은 점 4에서 주사선과 사귄다. 점 4에서의 세기를 얻는 빠른 방법은 주사선의 수직변위만을 리용하여 세기 I_1와 I_2사이에서 보간하는것이다.

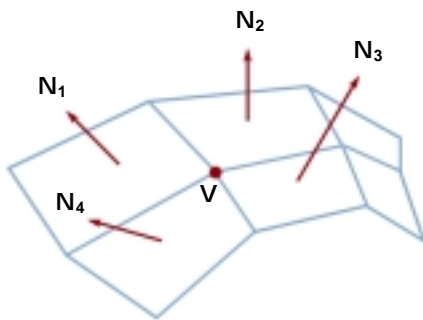



그림 14-44. 정점 **V**에서의 법선벡토르는 그 정점을 공유하는 매개 다각형에 대한 면의 법선들의 평균으로 계산된다.

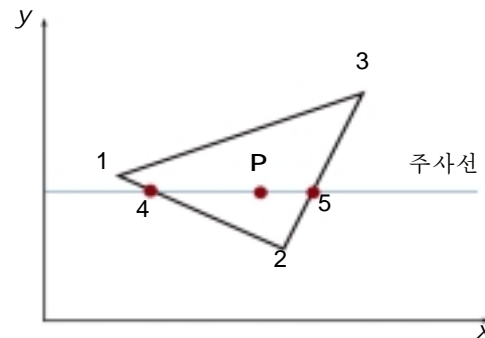



그림 14-45. 그로우명암처리에서 점 4의 세기는 정점 1과 2의 세기로부터 선형보간된다. 점 5의 세기는 정점 2와 3의 세기로부터 선형보간된다. 그러면 내부점 **P**에는 위치 4와 5의 세기로부터 선형보간된 세기값이 할당된다.

$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2 \qquad (14-38)$$

류사하게 이 주사선의 오른쪽 사귐점(점5)에서의 세기는 정점 2와 3의 세기값으로부터 보간된다. 이 테두리세기들이 주사선에 대하여 설정되면 내부점은(그림 14-45에서 점 **P**와 같은) 점 4와 5에서의 테

두리세기로부터

$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5 \qquad (14-39)$$

와 같이 보간된다.

주사선들사이에서 련속되는 변세기값을 얻는데 그리고 주사선을 따라 련속하는 세기를 얻는데 증분계산을 리용한다. 그림 14-46에 보여 준바와 같이 변우의 위치 (x, y)에서의 세기가

$$I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_2 \qquad (14-40)$$

과 같이 보간되면 이 변을 따라 다음 주사선에 대한 세기는

$$I' = I + \frac{I_2 - I_1}{y_1 - y_2} \qquad (14-41)$$

과 같이 얻을수 있다. 류사한 계산은 매개 주사선을 따라 련속하는 수평화소위치들에서의 세기를 얻는데 리용된다.

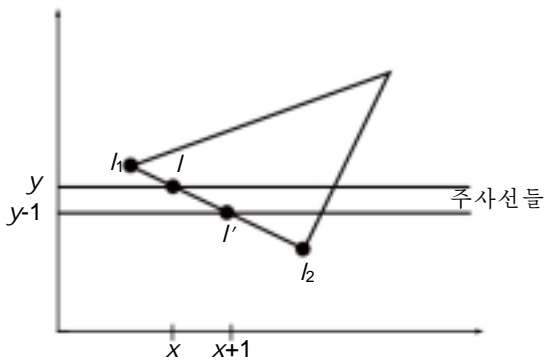



그림 14-46. 련속주사선들에서 다각형의 변을 따라 세기값들을 증분보간

면이 색실감처리될 때 정점들에서 매개 색성분의 세기가 계산된다. 그로우명암처리는 매개 주사선을 따라 보이는 다각형을 채우기 위하여 보이지 않는 면알고리듬과 결합될수 있다. 그로우방법에 의하여 명암처리된 물체의 실례를 그림 14-47에서 보여 주었다.

그로우명암처리는 단색명암처리에서의 세기의 불련속성은 제거하지만 몇가지 다른 결함들을 가진다. 면에서 광채는 때때로 변태적인 형태로 현시되며 선형세기보간은 마흐띠라고 하는 밝거나 어두운 세기의 줄무늬가 면에 나타나게 할수 있다. 이 효과들은 면을 더 많은 개수의 다각형면들로 분할하거나 또는 더 많은 계산을 요구하는 퐁명암처리와 같은 다른 방법들을 리용하여 줄일수 있다.

ㄱ)

ㄴ)

ㄷ)

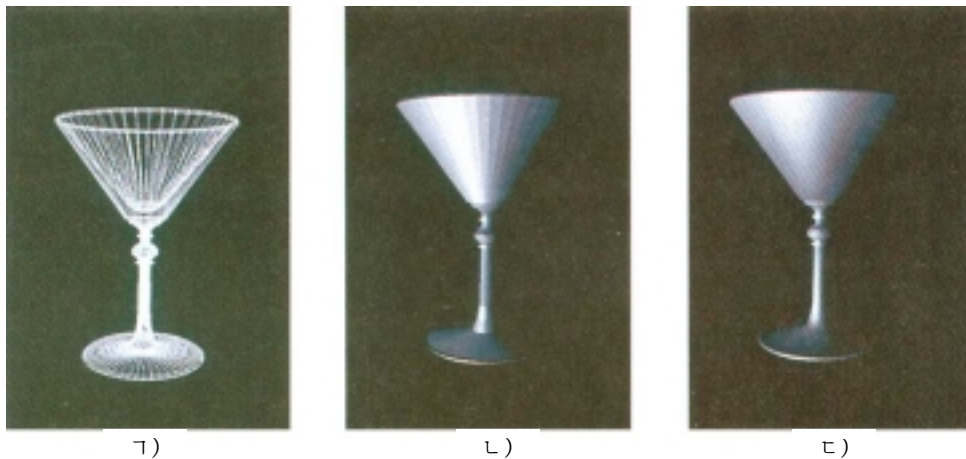



그림 14-47. 물체의 다각형그물근사(ㄱ)는 단색명암처리(ㄴ)와 그로우명암처리(ㄷ)에 의하여 실감처리된다.

퐁명암처리

다각형면을 실감처리하는 보다 정밀한 방법은 법선벡토르들을 보간하고 다음에 면우의 매점에 조명모형을 적용하는것이다. 이 방법은 Phong Bui Tuong에 의하여 개발되였으며 **퐁명암처리**(Phong shading) 또는 법선벡토르의 보간명암처리(normal-vector interpolation shading)라고 한다. 그것은 면에서 보다 현실감 있는 광채를 현시하며 마흐띠효과를 크게 줄인다.

다각형면은 퐁명암처리에서 다음의 걸음들을 수행하여 실감처리된다.

- 다각형의 매 정점에서 평균단위법선벡토르를 결정한다.
- 다각형의 면우에서 정점의 법선들을 선형보간한다.
- 매개 주사선을 따라 면우의 점에 대한 투영된 화소세기를 계산하는데 조명모형을 적용한다.

두 정점사이의 다각형변을 따라 면의 법선들을 보간하는것을 그림 14-48에서 보여 주었다. 정점 1과 2사이의 변을 따라 주사선과의 사귐점에 대한 법선벡토르 **N**은 변의 끝점들의 법선들사이를 수직으로 보간하여 얻을수 있다.

$$\mathbf{N} = \frac{y - y_2}{y_1 - y_2}\mathbf{N}_1 + \frac{y_1 - y}{y_1 - y_2}\mathbf{N}_2 \qquad (14-42)$$

증분방법은 주사선들사이와 매개 개별적인 주사선을 따라 법선들을 계산하는데 리용된다. 주사선을 따라 매개 화소위치에서 조명모형은 그 점에서의 면의 세기를 결정하는데 적용된다.

주사선을 따라 매개 점에서 근사된 법선벡토르를 리용하는 세기계산은 그로우명암처리에서와 같이 세기들을 직접 보간하는것보다 더 정밀한 결과를 만든다. 반면에 퐁명암처리는 현저하게 더 많은 계산을 요구한다.

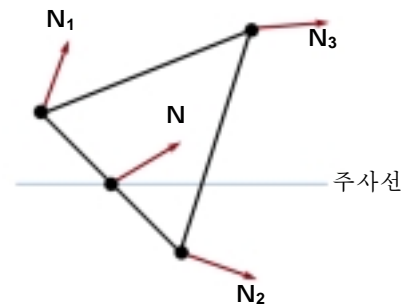



그림 14-48. 다각형의 변을 따라 면법선들을 보간

고속퐁명암처리

퐁명암처리에 의한 면실감처리는 조명모형계산에서 법선벡토르의 근사를 리용하면 속도를 높일수 있다. **고속퐁명암처리**(fast Phong shading)는 테일러합렬전개와 삼각형면쪼각을 리용하여 세기계산을 근사한다.

퐁명암처리에서는 정점의 법선들로부터 법선벡토르를 보간하기때문에 삼각형우의 임의의 점(x, y)에서의 면의 법선 **N**은

$$\mathbf{N} = \mathbf{A}x + \mathbf{B}y + \mathbf{C} \qquad (14-43)$$

과 같이 표현할수 있다. 여기서 벡토르 **A**, **B**, **C**는 세개의 정점방정식

$$\mathbf{N}_k = \mathbf{A}x_k + \mathbf{B}y_k + \mathbf{C}, \quad k = 1,2,3 \qquad (14-44)$$

로부터 결정된다. 여기서 (x_k, y_k)는 정점의 위치를 표시한다.

반사률과 감쇠파라메터들을 없애면 면우의 점(x, y)에서 광원의 확산반사에 대한 계산은 다음과 같다.

$$\begin{aligned} I_{diff}(x,y) &= \frac{\mathbf{L}\cdot\mathbf{N}}{|\mathbf{L}|\,|\mathbf{N}|} \\ &= \frac{\mathbf{L}\cdot(\mathbf{A}x + \mathbf{B}y + \mathbf{C})}{|\mathbf{L}|\,|\mathbf{A}x + \mathbf{B}y + \mathbf{C}|} \qquad (14-45) \\ &= \frac{(\mathbf{L}\cdot\mathbf{A})x + (\mathbf{L}\cdot\mathbf{B})y + \mathbf{L}\cdot\mathbf{C}}{|\mathbf{L}|\,|\mathbf{A}x + \mathbf{B}y + \mathbf{C}|} \end{aligned}$$

이 식을

$$I_{\text{diff}}(x,y)=\frac{ax+by+c}{(dx^2+exy+fy^2+gx+hy+i)^{1/2}} \tag{14-46}$$

의 형식으로 다시 쓸수 있다. 여기서 a, b, c, d와 같은 파라메터들은 여러가지 스칼라적들을 표현하는데 리용된다. 실례로

$$a=\frac{\mathbf{L}\cdot\mathbf{A}}{|\mathbf{L}|} \tag{14-47}$$

이다. 마지막으로 식 14-46의 분모를 테일러합렬전개로 표현할수 있으며 x와 y의 2차까지의 항들을 보유할수 있다. 이것은

$$I_{\text{diff}}(x,y)=T_5x^2+T_4xy+T_3y^2+T_2x+T_1y+T_0 \tag{14-48}$$

을 낳는다. 여기서 매 T_k는 파라메터 a, b, c 등의 함수이다.

앞계차를 리용하면 초기앞계차파라메터들이 계산된후 매 화소위치 (x, y)에 대한 두개의 더하기만으로 식 14-48을 계산할수 있다. 고속퐁명암처리는 퐁명암처리계산을 줄이지만 고속퐁명암처리에 의하여 면을 실감처리하는것은 그로우명암처리에 의하여 하는것보다 여전히 약 2배나 더 길다. 앞계차를 리용하는 표준퐁명암처리는 그로우명암처리보다 약 6～7배 더 길다.

확산반사에 대한 고속퐁명암처리는 거울반사를 포함하도록 확장할수 있다. 확산반사와 류사한 계산들은 기본조명모형에서의 $(\mathbf{N}\cdot\mathbf{H})^{n_s}$와 같은 거울항들을 계산하는데 리용된다. 게다가 3각형이 아닌 다각형과 유한보기위치를 포함하도록 알고리듬을 일반화할수 있다.

6절. 광선추적방법

10장 15절에서는 광선투사개념을 소개하였다. 광선투사에서는 립체구성기하방법들을 리용하여 물체를 모형화할 때 면사귐점들을 알아 내기 위하여 매 화소위치로부터 광선을 내보낸다. 또한 장면에서 보이는 면들을 결정하기 위한 한가지 방법으로서 광선투사의 리용을 론의하였다(13장 10절). **광선추적**(ray tracing)은 이 방법의 확장이다. 단순히 매 화소에 대하여 보이는 면을 찾는 대신에 그림 14-49에서 보여 주는 바와 같이 세기몫들을 더하면서 장면에서 광선이 계속 튀여 나게 한다. 이것은 대역반사와 투과효과를 얻기 위한 간단하면서도 위력한 실감처리기법을 제공한다. 또한 기본광선추적알고리듬은 보이는 면의 검출, 그림자효과, 투명효과, 다중광원조명효과에 적용된다. 사진과 같이 현시하기 위하여 기본알고리듬에 대한 많은 확장들이 개발되였다. 광선추적된 현시

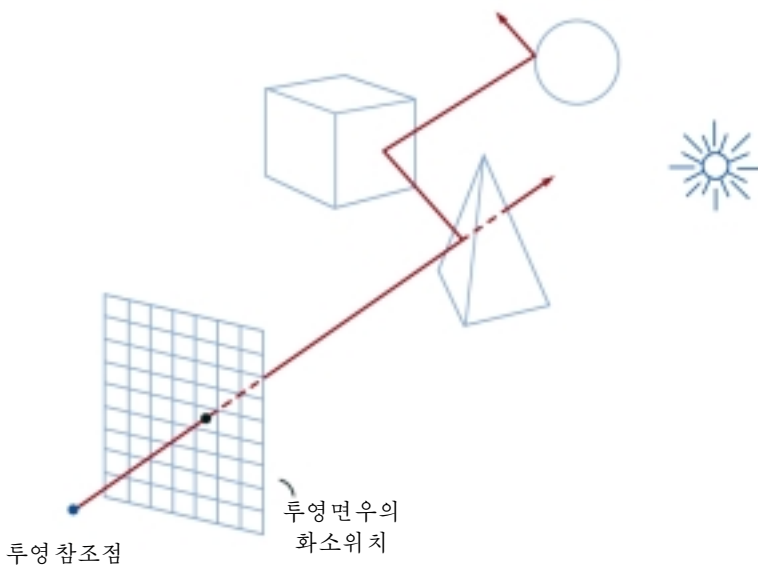



그림 14-49. 투영참조점으로부터 화소위치를 지나 광선을 추적할 때의 다중반사와 투과

그는 특히 광택이 있는 물체에서 대단히 현실감이 있지만 그것을 만드는데 상당한 계산시간이 요구된다. 광선추적에서 가능한 대역반사와 투파효과의 실례를 그림 14-50에 보여 주었다.

그림 14-50. 물체면들에서의 대역반사와 투파조명효과를 보여 주는 광선추적된 장면

광선추적의 기본알고리듬

먼저 *xy*평면에서 지적되는 화소위치들에 의하여 자리표계를 설정한다. 장면표현은 이 자리표계에서 주어 진다(그림 14-51). 투영중심으로부터 매개 화면 화소위치의 중심을 지나는 광선경로를 결정한다. 이 광선경로를 따라 축적된 조명효과는 화소에 할당된다. 이 실감처리방법은 기하광학원리에 기초한다. 장면안의 면들로부터의 광선은 모든 방향으로 퍼지며 일부는 투영평면의 화소위치들을 지날것이다. 무한개의 광선경로가 있기때문에 화소로부터 장면에로의 빛경로를 추적하여 개개 화소에 대한 기여들을 결정한다. 먼저 화소당 한개의 광선을 리용하여 기본광선추적알고리듬을 고찰한다. 이것은 바늘구멍카메라로 장면을 보는것과 같다.

매개 화소광선에 대하여 장면안의 매개 면이 그 광선에 의해 가로질러 지는가를 검사한다. 면이 가로질러지면 화소로부터 면의 사귐점까지의 거리를 계산한다. 계산된 제일 작은 사귐점거리는 그 화소에 대한 보이는 면을 식별한다. 다음에 광선을 보이는 면으로부터 거울경로(입사각과 같은 반사각)를 따라 반사시킨다. 또한 면이 투명하면 광선을 면을 통하여 굴절방향으로 보낸다. 반사 및 굴절광선을 2차광선이라고 한다.

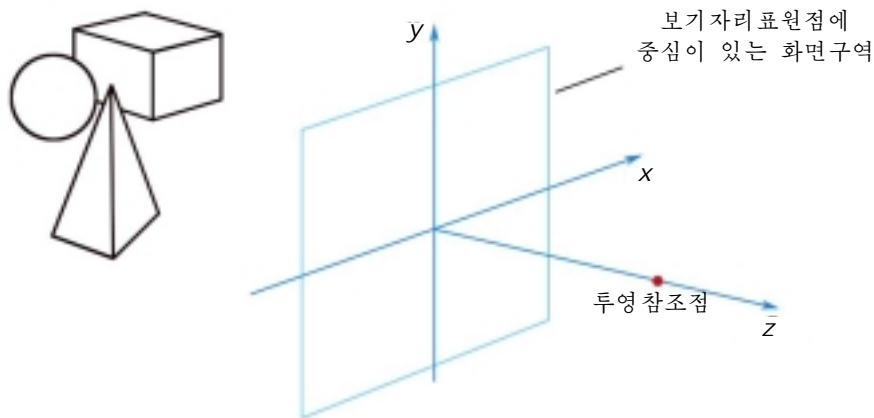



그림 14-51. 광선추적자리표계

이 절차는 매개 2차광선에 대하여 반복된다. 즉 광선과 물체들과의 사귐을 검사하고 2차광선경로를 따라 제일 가까운 면은 다음번 반사 및 굴절경로를 재귀적으로 발생시키는데 리용된다. 화소로부터의 광선이 장면을 따라 스쳐 지날 때 매개 련속적으로 가로질러지는 면은 그림 14-52에 보여 준바와 같이 2분광선추적나무에 추가된다. 나무의 왼쪽 가지는 반사경로를 표현하는데 리용하며 오른쪽 가지는 투과경로를 표현한다. 광선추적나무의 최대깊이는 사용자선택항목으로 설정될수 있으며 또는 사용가능한 기억기량에 의하여 결정될수 있다. 그러면 나무의 경로는 그것이 미리 설정된 최대값에 도달하거나 또는 광선이 광원과 마주칠 때 끝난다.

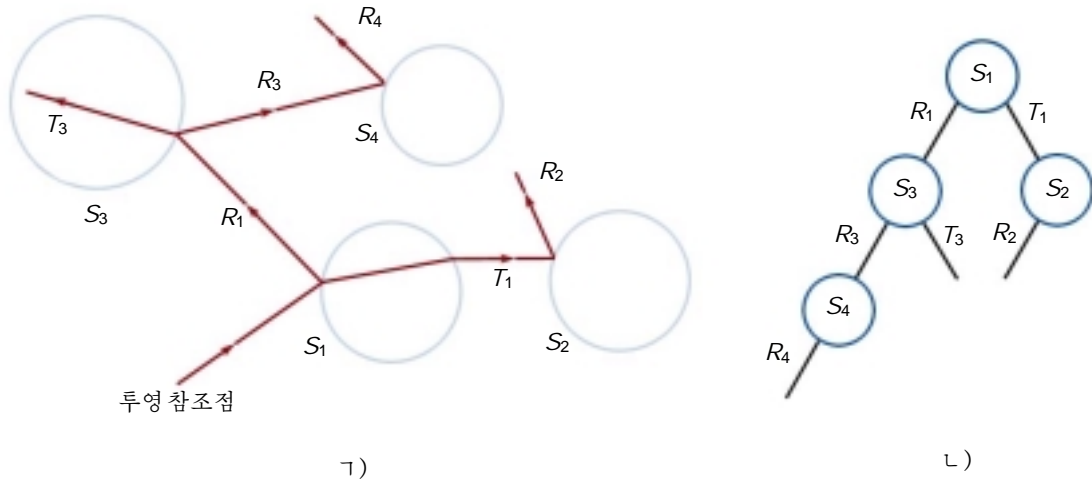



그림 14-52. ㄱ-화면화소에서 장면을 통한 반사 및 굴절광선경로,
ㄴ-ㄱ에 보여 준 경로들에 대한 2분광선추적나무

화소에 할당되는 세기는 광선추적나무의 밑(말단마디)에서부터 세기몫들이 축적되여 결정된다. 나무의 매 마디의 면의 세기는 어미면(나무의 웃쪽 다음 마디)으로부터 이 면까지의 거리에 의하여 감쇠되며 어미면의 세기에 추가된다. 그러면 화소의 세기는 광선나무의 뿌리마디에서의 감쇠된 세기들의 합이다. 화소광선에 의하여 가로질러 지는 면이 없으면 광선추적나무는 비고 그 화소에는 배경세기값이 할당된다. 비록 광원은 보통 초기광선경로의 뒤에 있지만 화소광선이 비반사광원을 가로지르면 그 화소에는 광원의 세기를 할당할수 있다.

그림 14-53에서는 광선에 의하여 가로질러 지는 면과 반사빛세기계산에 필요한 단위벡토르들을 보여 주었다. 단위벡토르 **u**는 광선경로방향이며 **N**은 면의 단위법선, **R**는 단위반사벡토르, **L**은 광원을 가리키는 단위벡토르, **H**는 **V**(**u**의 반대)와 **L**사이의 중간단위벡토르이다. **L**을 따르는 경로는 그림자광선이라고 한다. 어떤 물체가 면과 점광원사이의 그림자광선을 가로지르면 그 면은 그 광원에 대한 그림자구역에 있다. 면에서 주변빛은 $k_a I_a$와 같이 계산되며 광원으로 인한 확산반사는 $k_d(\mathbf{N}\cdot\mathbf{L})$에 비례하고 거울반사성분은 $k_s(\mathbf{H}\cdot\mathbf{N})^{n_s}$에 비례한다. 14장 2절에서 설명한바와 같이 2차광선경로 **R**에 대한 거울반사방향은 면의 법선과 입사광선방향에 관계된다.

$$\mathbf{R} = \mathbf{u} - (2\mathbf{u}\cdot\mathbf{N})\mathbf{N} \tag{14-49}$$

또한 투명한 면일 때에는 재료를 통하여 투과되는 빛으로부터의 세기몫을 얻어야 한다. 이 광원은 그림 14-54에 보여 준바와 같이 투과방향 **T**를 따라 2차광선을 추적하는 방법으로 찾아 낼수 있다. 단위투과벡토르는 벡토르 **u**와 **N**으로부터

$$\mathbf{T} = \frac{\eta_i}{\eta_r}\mathbf{u} - \left(\cos\theta_r - \frac{\eta_i}{\eta_r}\cos\theta_i\right)\mathbf{N} \tag{14-50}$$

과 같이 얻을수 있다. 파라메터 η_i와 η_r는 각각 입사재질과 굴절재질의 굴절률이다. 굴절각 θ_r는 스넬의 법칙으로부터 계산할수 있다.

$$\cos\theta_r = \sqrt{1 - \left(\frac{\eta_i}{\eta_r}\right)^2 (1 - \cos^2\theta_i)} \tag{14-51}$$

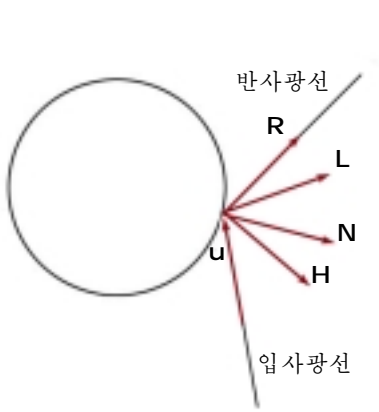



그림 14-53. 방향 **u** 를 따라 입사하는 광선에 의하여 가로질러 지는 물체면에서의 단위벡토르들

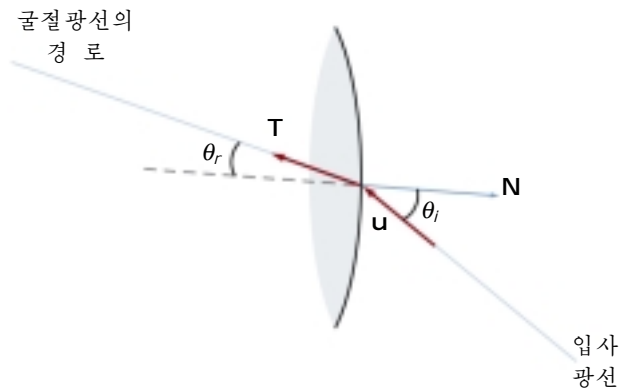



그림 14-54. 투명한 재질을 통한 굴절광선경로 **T**

광선-면사귐점계산

광선은 그림 14-55에서 보여 주는바와 같이 초기위치 $\mathbf{P}_0$과 단위방향벡토르 **u**에 의하여 표현할수 있다. 광선을 따라 $\mathbf{P}_0$으로부터 거리 s에 있는 임의의 점 **P**의 자리표는 광선의 방정식

$$\mathbf{P} = \mathbf{P}_0 + s\,\mathbf{u} \qquad (14-52)$$

로부터 계산된다. 초기에 $\mathbf{P}_0$은 투명면우의 화소위치로 설정될수 있으며 또는 투영참조점으로 선택될수 있다. 단위벡토르 **u**는 초기에 광선이 통과하는 화소의 위치와 투영참조점으로부터 얻어 진다.

$$\mathbf{u} = \frac{\mathbf{P}_{\text{pix}} - \mathbf{P}_{\text{prp}}}{\left|\mathbf{P}_{\text{pix}} - \mathbf{P}_{\text{prp}}\right|} \qquad (14-53)$$

매개 가로질러 지는 면에서 벡토르 P_0과 u는 광선-면사귐점에서의 2차광선에 대하여 갱신된다. 2차광선에서 **u**에 대한 반사방향은 **R**이고 투과방향은 **T**이다. 면의 사귐점을 찾아 내기 위하여 광선의 방정식과 장면안의 개별적인 물체에 대한 면의 방정식을 련립하여 푼다.

광선추적에서 제일 간단한 물체는 구이다. 반경이 r이고 중심위치가 $\mathbf{P}_c$인 구(그림 14-56)를 가지면 면우의 임의의 점 **P**는 구의 방정식

$$\left|\mathbf{P} - \mathbf{P}_c\right|^2 - r^2 = 0 \qquad (14-54)$$

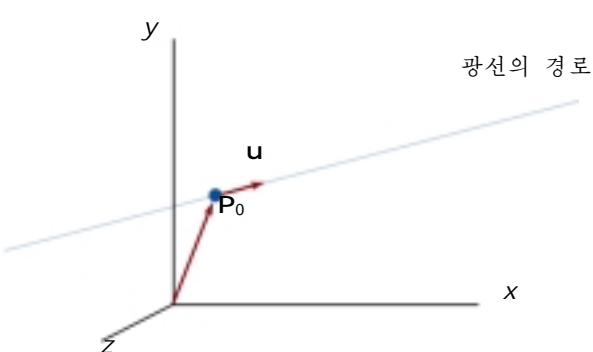



그림 14-55. 초기위치벡토르 $\mathbf{P}_0$과 단위 방향벡토르 **u**에 의한 광선의 표현

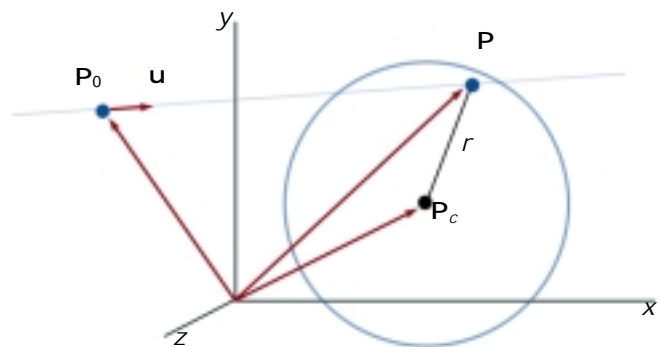



그림 14-56. 반경이 r이고 중심이 위치 $\mathbf{P}_c$에 있는 구를 가로지르는 광선

을 만족시켜야 한다. 광선의 방정식 14-52를 대입하면

$$|\mathbf{P}_0 + s\mathbf{u} - \mathbf{P}_c|^2 - r^2 = 0 \qquad (14-55)$$

를 가진다. $\Delta\mathbf{P} = \mathbf{P}_c - \mathbf{P}_0$이라고 하고 스칼라적을 전개하면 2차방정식

$$s^2 - 2(\mathbf{u}\cdot\Delta\mathbf{P})s + (|\Delta\mathbf{P}|^2 - r^2) = 0 \qquad (14-56)$$

을 얻으며 그의 풀이는

$$s = \mathbf{u}\cdot\Delta\mathbf{P} \pm \sqrt{(\mathbf{u}\cdot\Delta\mathbf{P})^2 - |\Delta\mathbf{P}|^2 + r^2} \qquad (14-57)$$

이다. 판별식이 부이면 광선은 구를 가로지르지 않는다. 그렇지 않으면 면사귐점자리표는 광선의 방정식 14-52로부터 식 14-57의 둘중 더 작은 값을 리용하여 얻어 진다.

초기광선위치로부터 멀리에 있는 작은 구인 경우 식 14-57은 둥그리기오차에 민감하다. 즉

$$r^2 << |\Delta\mathbf{P}|^2$$

이면 $|\Delta\mathbf{P}|^2$의 정확성오차에서 r^2항을 잃을수 있다. 대부분의 경우들에 이것은 거리 s에 대한 계산을

$$s = \mathbf{u}\cdot\Delta\mathbf{P} \pm \sqrt{r^2 - |\Delta\mathbf{P} - (\mathbf{u}\cdot\Delta\mathbf{P})\mathbf{u}|^2} \qquad (14-58)$$

과 같이 재정돈하여 피할수 있다. 대역적인 면반사를 현시하기 위하여 그림 14-57에서는 광선추적에 의하여 실감처리된 광택나는 구들의 눈송이무늬를 보여 주었다.

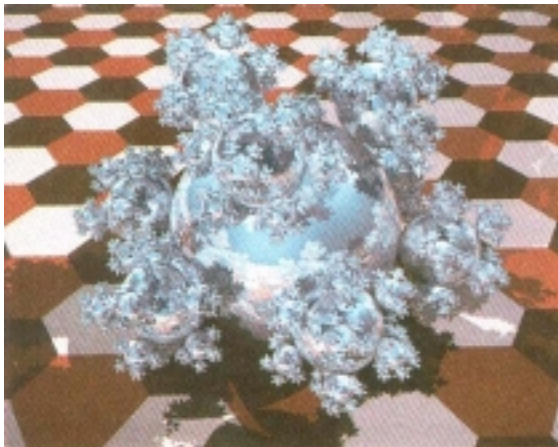



그림 14-57. 7381개의 구와 3개의 광원을 리용한 광선추적에 의하여 실감처리된 《구송이》

다면체는 면사귐점을 찾아 내는데 구보다 더 많은 처리를 요구한다. 이런 리유로 하여 경계체적에서 초기사귐점검사를 하는것이 더 좋다. 실례로 그림 14-58은 구에 의하여 경계 지어 지는 다면체를 보여 준다. 광선이 구를 가로지르지 않으면 그 다면체에서 앞으로 임의의 검사를 할 필요가 없다. 그러나 광선이 구를 가로지르면 먼저 검사

$$\mathbf{u}\cdot\mathbf{N} < 0 \qquad (14-59)$$

에 의해 앞면을 찾아 낸다. 여기서 $\mathbf{N}$은 면의 법선이다. 부등식 14-59를 만족시키는 다면체의 매면에 대하여 평면의 방정식

$$\mathbf{N}\cdot\mathbf{P} = -D \qquad (14-60)$$

을 광선의 방정식 14-52를 만족시키는 면의 위치 $\mathbf{P}$에 대하여 푼다. 여기서 $\mathbf{N}=(A, B, C)$이고 D는 평면의 4번째 파라메터이다. 위치 $\mathbf{P}$는

$$\mathbf{N}\cdot(\mathbf{P}_0+s\mathbf{u})=-D \qquad (14-61)$$

이면 평면과 광선경로에 다같이 있다. 초기광선위치로부터 평면까지의 거리는

$$s=-\frac{D+\mathbf{N}\cdot\mathbf{P}_0}{\mathbf{N}\cdot\mathbf{u}} \qquad (14-62)$$

이다. 이것은 다각형면을 포함하는 무한평면에서의 위치를 주며 그러나 이 위치는 다각형경계안에 있지 않을수 있다(그림 14-59). 그리하여 광선이 다면체의 이 면을 가로지르는가를 결정하기 위하여 내부-외부검사(3장)를 할 필요가 있다. 이 검사는 부등식 14-59를 만족시키는 매면에 대하여 수행한다. 내부점까지의 제일 작은 거리 s는 가로질러 지는 다면체의 면을 식별한다. 식 14-62로부터의 사귐점위치가 내부점이 아니면 광선은 물체를 가로지르지 않는다.

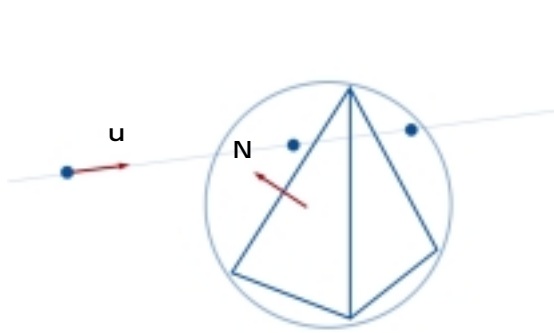



그림 14-58. 경계구에 의하여 둘러 싸이는 다면체

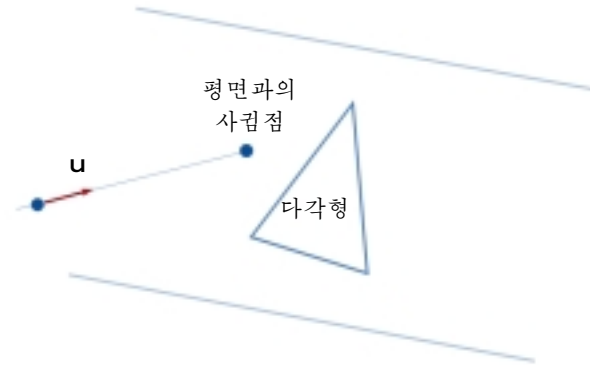



그림 14-59. 다각형의 평면과 광선과의 사귐

류사한 절차는 2차 또는 스플라인곡면과 같은 다른 물체들에 대하여 광선-면사귐점위치를 계산하는데 리용된다. 광선의 방정식을 곡면정의와 결합하고 파라메터 s에 대하여 푼다. 많은 경우 수값풀이구하기방법과 증분계산이 면에서의 사귐점위치를 찾아 내는데 리용된다. 그림 14-60에서는 다중물체와 걸문양을 포함하는 광선추적된 장면을 보여 주었다.

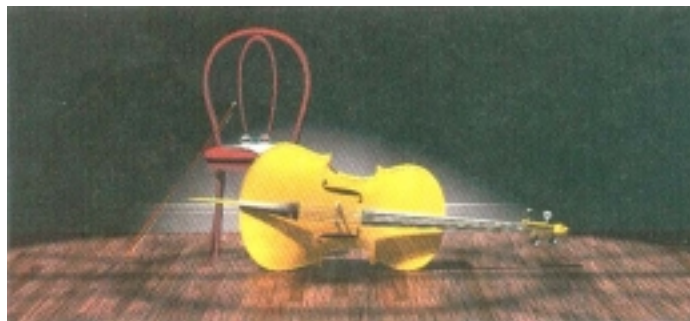



그림 14-60. 면의 걸문양에서 대역적인 반사를 보여 주는 광선추적된 장면

물체사귐점계산의 줄이기

광선추적알고리듬에서 광선-면사귐점계산은 전체 처리시간의 95%정도를 차지한다. 물체들이 많은 장면에서 매개 광선에 대한 대부분의 처리시간은 광선경로를 따라 보이지 않는 물체들을 조사하는데 소비된다. 그러므로 이 사귐점계산에 소비되는 처리시간을 줄이기 위한 여러가지 방법들이 개발되였다.

사귐점계산을 줄이는 한가지 방법은 린접한 물체들을 구나 직6면체와 같은 경계체적으로 둘러

싸고(그림 14-61) 광선-면사귐검사에 경계체적을 리용하는것이다. 광선이 경계물체와 사귀지 않으면 둘러 싸이는 면들과의 사귐검사를 없앨수 있다. 이 방법은 경계체적의 계층을 포함하도록 확장될수 있다. 즉 여러가지 경계체적들을 더 큰 체적으로 둘러 싸고 사귐검사를 계층적으로 수행한다. 먼저 바깥경계체적을 검사하고 필요하면 더 작은 내부경계체적을 검사한다.

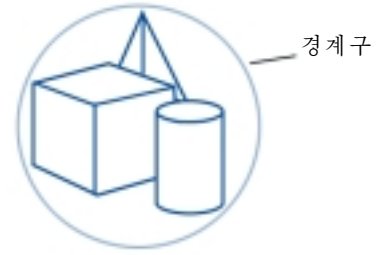



그림 14-61. 경계구로 둘러 싼 물체들

공간부분분할방법

사귐점계산을 줄이는 다른 방법은 **공간부분분할방법**(space-subdivision method)을 리용하는것이다. 장면을 바른6면체로 둘러 싸고 그의 매개 부분공간(세포)안에 면들이 이미 설정된 최대개수보다 많지 않을 때까지 바른6면체를 련속적으로 부분분할한다. 실례로 매개 세포안에 한개의 면만 있도록 요구할수 있다. 병렬 및 벡토르처리능력이 사용가능하면 세포당 면들의 최대개수는 벡토르등록기의 크기와 처리소자의 개수에 의하여 결정할수 있다. 바른6면체의 공간부분분할은 8분나무 또는 2분공간분할나무에 기억될수 있다. 게다가 바른6면체를 매 걸음에서 크기가 같은 8개의 8분공간들로 나누어 균등부분분할(uniform subdivision)할수 있으며 또는 적응부분분할(adaptive subdivision)하고 물체를 포함하는 바른6면체의 부분공간들만을 부분분할할수 있다.

다음에 바른6면체의 개별적인 세포들을 따라 광선을 추적하며 면을 가지는 세포안에서만 사귐검사를 한다. 광선이 가로지르는 첫번째 물체면은 그 광선에 대하여 보이는 면이다. 세포의 크기와 세포당 면의 개수사이에 균형이 이루어 진다. 세포당 면들의 최대개수를 너무 작게 설정하면 세포의 크기는 줄여 진 사귐검사에서의 많은 보관들이 세포가로지르기처리에 들어 가도록 작게 될수 있다.

그림 14-62에서는 화소광선과 장면을 둘러 싸는 바른6면체의 앞면과의 사귐을 보여 주었다. 바른6면체의 앞면에서의 사귐점을 계산하면 사귐점의 자리표를 세포들의 경계위치에서 검사하는 방법으로 초기의 사귐세포를 결정한다. 다음에 물체의 면을 가로지르거나 또는 장면을 둘러 싸는 바른6면체를 벗어 날 때까지 광선에 의하여 가로질러지는 매개 세포에 대한 입구 및 출구점(그림14-63)을 결정하여 세포들을 따라 광선을 처리한다.

광선의 방향 $\mathbf{u}$와 세포에 대한 광선입구위치 $\mathbf{P}_{in}$이 주어 지면 가능한 출구면은

$$\mathbf{u} \cdot \mathbf{N}_k > 0 \qquad (14-63)$$

인것들이다. 그림 14-63의 세포면들에 대한 법선벡토르들이 자리표축과 정렬되면

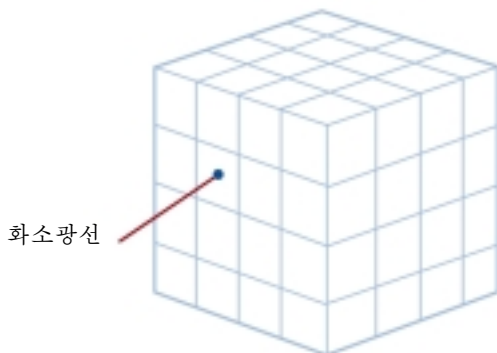



그림 14-62. 장면안의 모든 물체들을 둘러 싸는 바른6면체와 광선과의 사귐

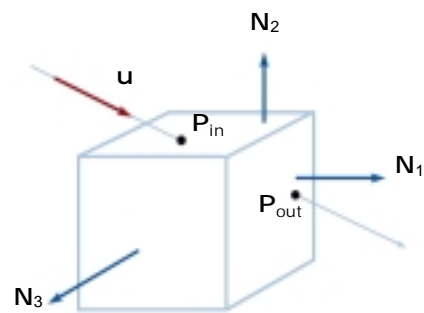



그림 14-63. 장면을 둘러 싸는 바른6면체의 부분공간(세포)을 가로지르는 광선

$$\mathbf{N}_k = \begin{cases} (\pm1,0,0) \\ (0,\pm1,0) \\ (0,0,\pm1) \end{cases}$$

이고 3개의 후보출구면을 결정하자면 **u**의 매개 성분의 부호만 검사하면 된다. 매개 후보면에서의 출구위치는 광선의 방정식

$$\mathbf{P}_{\text{out},k} = \mathbf{P}_{\text{in}} + s_k \mathbf{u} \qquad (14-64)$$

로부터 얻어 진다. 여기서 s_k는 광선을 따라 $\mathbf{P}_{\text{in}}$으로부터 $\mathbf{P}_{\text{out},k}$까지의 거리이다. 광선의 방정식을 매개 세포면에 대한 평면의 방정식에 대입하면

$$\mathbf{N}_k \cdot \mathbf{P}_{\text{out},k} = -D \qquad (14-65)$$

이고 매개 후보출구면까지의 광선거리에 대하여

$$s_k = \frac{-D - \mathbf{N}_k \cdot \mathbf{P}_{\text{in}}}{\mathbf{N}_k \cdot \mathbf{u}} \qquad (14-66)$$

과 같이 풀수 있다. 이 계산은 법선벡토르 $\mathbf{N}_k$가 자리표축과 정렬되면 간단해 질수 있다. 실례로 후보 법선벡토르가 (1, 0, 0)이면 그 평면에 대하여

$$s_k = \frac{x_k - x_0}{u_x} \qquad (14-67)$$

이다. 여기서 $\mathbf{u}=(u_x, u_y, u_z)$이고 x_k는 세포의 오른쪽 경계면의 값이다.

처리속도를 높이기 위하여 세포가로지르기절차를 여러가지로 수정할수 있다. 한가지 가능성은 시험출구면 k를 **u**의 제일 큰 성분의 방향에 수직인것으로 취하는것이다. $\mathbf{P}_{\text{out},k}$를 포함하는 시험면에서 구역은 진짜 출구면을 결정한다(그림14-64). 사귐점$\mathbf{P}_{\text{out},k}$가 구역 0에 있으면 시험면은 진짜 출구면이고 여기서 끝낸다. 사귐점이 구역 1에 있으면 진짜 출구면은 웃면이고 세포의 웃경계에서 출구점을 간단히 계산한다. 류사하게 구역 3은 아래면을 진짜 출구면으로 식별하고 구역 4와 2는 진짜 출구면을 각각 왼쪽 및 오른쪽 세포면으로 식별한다. 시험출구점이 구역 5, 6, 7, 8에 있을 때에는 진짜 출구면을 식별하기 위하여 2개의 추가적인 사귐점을 계산하여야 한다. 병렬벡토르기계에서 이 방법의 실현은 수행에서의 이 이상의 개선을 제공한다.

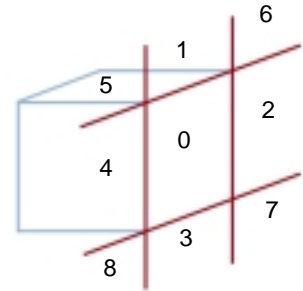



그림 14-64. 시험출구면의 구역들

그림 14-65의 장면은 공간부분분할방법을 리용하여 광선추적되였다. 공간부분분할이 없으면 광선추적계산은 10배 더 오래 걸린다. 또한 다각형들을 없애면 처리속도를 높일수 있다. 2048개의 구를

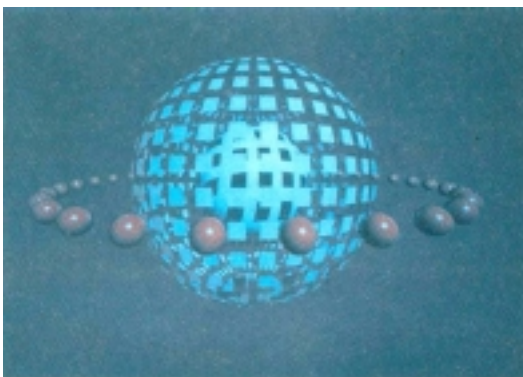



그림 14-65. 37개의 구와 720개의 다각형면을 포함하는 병렬광선추적된 장면(광선추적알고리듬은 화소당 9개의 광선과 나무깊이 5를 리용하였다. 공간부분분할방법은 Alliant FX/8에서 광선추적의 기본알고리듬보다 10배 더 빨리 장면을 처리하였다.)

포함하며 다각형이 없는 장면에 대하여 이 알고리듬은 광선추적의 기본알고리듬보다 46배 더 빨리 실행되였다.

그림 14-66에서는 공간부분분할과 병렬처리방법을 리용하여 광선추적된 다른 장면을 보여 주었다. 로뎅의 《생각하는 사람》의 이 화상은 24s에 150만개이상의 광선으로 추적되였다.

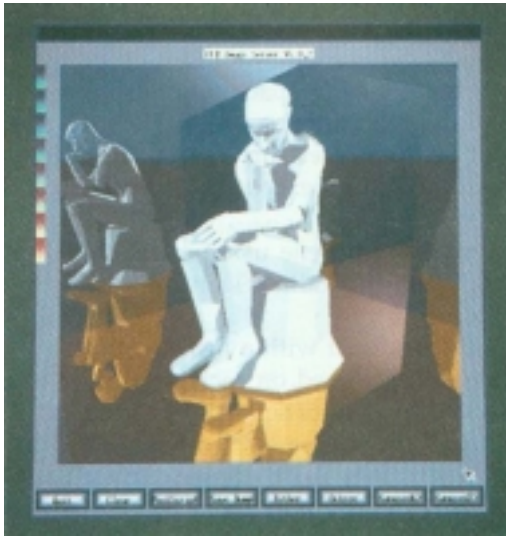



그림 14-66. 광선추적된 이 장면은 32개의 처리소자를 가지는 Kendall Square Research KSR1 병렬콤퓨터에서 실감처리하는데 24s 걸렸다(로뎅의 《생각하는 사람》은 3036개의 기초요소들로 모형화되였다. 2개의 광원과 화소당 1개의 1차광선을 리용하여 1675776개의 광선들을 처리함으로써 대역적인 조명효과를 얻었다.).

그림 14-67에 보여 준 장면은 빛완충기기술 즉 공간분할형식으로 실감처리되였다. 여기서 바른6면체는 매개 점광원에 중심이 있으며 바른6면체의 매개 측면은 바른4각형격자에 의하여 갈라 진다. 매개 바른4각형을 지나는 빛에 보이는 물체들의 정렬된 목록은 그림자광선의 처리속도를 높이기 위하여 광선추적알고리듬에 의해 유지된다. 면조명효과를 결정하기 위하여 매개 그림자광선에 대한 바른4각형이 계산되며 다음에 그림자광선은 그 바른4각형에 대한 물체들의 목록에 대하여 처리된다.

ㄱ)

ㄴ)

그림 14-67. 5개의 광원에 의하여 조명되는 방장면(ㄱ)은 그림자광선을 처리하는 광선추적 및 빛완충기기술을 리용하여 실감처리되였다(ㄱ에 보여 준 방의 부분확대 ㄴ는 대역적인 조명효과를 설명한다. 방은 1298개의 다각형, 4개의 구, 76개의 원기둥, 35개의 2차곡면으로 모형화된다. 실감처리시간은 VAX 11/780에서 빛완충기를 리용하지 않을 때 602min이고 리용할 때 246min이였다.).

광선추적프로그람에서 사귐검사는 원추형광선묶음을 고찰하는 방향적인 부분분할절차를 써서 줄일수 있다. 매개 원추안에서 면들은 그림 14-68에서와 같이 깊이순서로 정렬시킬수 있다. 그러면 매개 광선에 대하여 그 광선이 속하는 원추안의 물체들만을 검사한다.

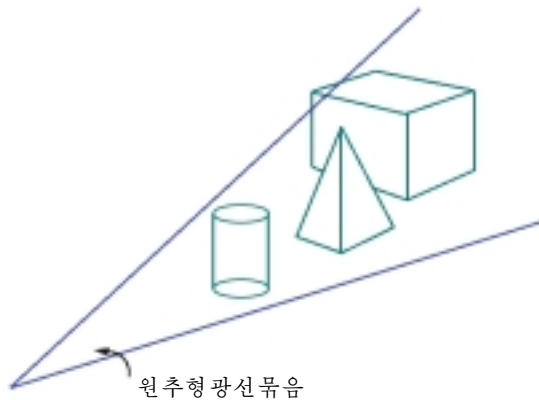



그림 14-68. 공간의 방향적인 부분분할(원추안의 모든 광선들에 대하여 원추안의 면들만을 깊이순서로 검사한다.)

경계허상제거를 고려한 광선추적

광선추적알고리듬에서 경계허상을 제거하는 두가지 기본기술은 초표본화와 적응표본화이다. 광선추적에서의 표본화는 4장에서 설명한 표본화방법들의 확장이다. 초표본화 및 적응표본화에서 화소는 단일한 점대신에 유한바른4각형으로 취급된다. 초표본화는 매개 화소구역에서 다중등간격광선(표본)을 리용한다. 적응표본화는 화소구역의 일부 부분구역에서 비등간격광선을 리용한다. 실례로 화소세기들을 더 잘 평가하기 위하여 물체의 변두리가까이에서 더 많은 광선들을 리용할수 있다. 표본화의 다른 방법은 화소구역우에서 광선들을 우연적으로 분포시키는것이다. 이 방법을 다음소제목에서 설명한다. 화소당 다중광선을 리용할 때 전체적인 화소세기를 만들기 위하여 화소광선들의 세기를 평균한다.

그림 14-69에서는 간단한 초표본화절차를 보여 주었다. 여기서는 화소의 매 구석을 통하여 한개의 광선이 발생된다. 4개 광선에 대한 세기들이 근사적으로 같지 않으면 또는 4개 광선사이에 어떤 작은 물체가 있으면 화소구역을 부분화소들로 나누고 처리를 반복한다. 실례로 그림 14-70의 화소는 9개의 부분화소들로 나뉘여 지며 매개 부분화소구석에서 한개씩 16개의 광선을 리용한다. 적응표본화는 거의 같은 세기의 광선들을 가지지 않거나 또는 어떤 작은 물체에 대하는 부분화소들을 앞으로 부분분할하는데 리용된다. 이 부분분할처리는 매개 부분화소가 거의 같은 세기의 광선들을 가지거나 또는 화소당 광선들의 개수가 웃한계 례컨대 256에 도달할 때까지 계속될수 있다.

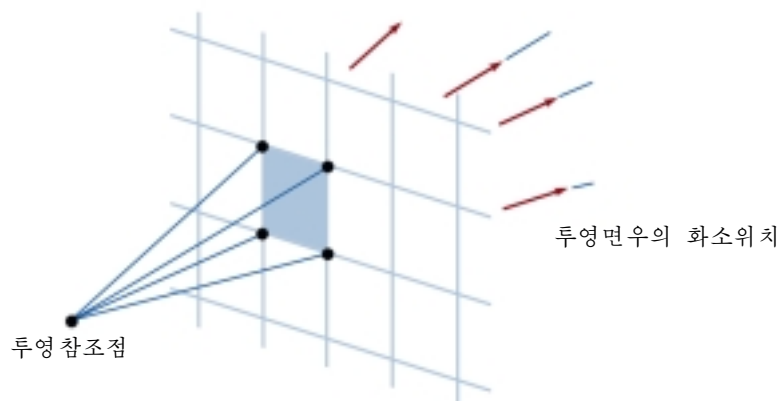



그림 14-69. 매개 화소구석에 하나씩 화소당 4 개 광선에 의한 표본화

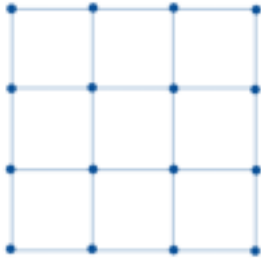



그림 14-70. 화소를 9개의 부분화소로 부분분할(부분화소의 매개 구석에 광선을 하나씩 둔다.)

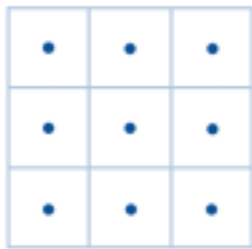



그림 14-71. 부분화소구역의 중심에 있는 광선위치

이 책의 겉표지그림은 Macintoshs II 의 Rayshade version3을 리용하여 적응부분분할광선추적방법으로 실감처리되였다. 확장광원은 현실감 있는 부드러운 그림자를 제공하는데 리용되였다. 약 2600만개의 1차광선들이 3350만개의 그림자광선, 6730만개의 반사광선과 함께 발생되였다. 나무의 결과 대리석면무늬는 잡음함수에 의해 립체결문양입히기방법을 리용하여 얻었다. 확장광원에 의한 총실감처리시간은 213시간이였다. 그림 2-20에 보여 준 립체쌍의 매개 화상은 점광원을 리용하여 45시간동안에 얻었다.

광선을 화소의 구석들을 통과시키는 대신에 그림 14-71에서와 같이 광선을 부분화소의 중심을 통과시켜 만들수 있다. 이 방법에서 광선들에는 4장에서 설명한 표본화계획들중 하나에 따라 무게를 달수 있다.

현시되는 장면의 경계허상을 제거하는 다른 방법은 그림 14-72에 보여 준바와 같이 화소광선을 원추로 취급하는것이다. 화소당 한개 광선만이 발생되지만 광선은 지금 유한사귐부분을 가진다. 물체에 대한 화소구역의 적용범위의 퍼센트를 결정하기 위하여 화소원추와 물체면과의 사귐을 계산한다. 이것은 구인 경우 두 원의 사귐을 찾아야 한다. 다면체일 때에는 원과 다각형의 사귐을 찾아야 한다.

분산광선추적

이 방법은 조명모형에서의 여러가지 파라메터들에 따라 광선을 우연적으로 분산시키는 확률표본화방법이다. 조명파라메터에는 화소의 면적, 반사 및 굴절방향, 카메라렌즈면적, 시간이 속한다. 따라서 경계허상효과는 낮은 준위잡음으로 교체되며 그것은 그림의 질을 개선하고 면의 광택과 투명도, 유한카메라시야조절장치, 유한광선, 움직이는 물체의 운동흐림현시를 보다 정밀하게 모형화할수 있게 한다. 분산된 광선추적(distributed ray tracing)이라고도 하는 분산광선추적(distribution ray tracing)은 본질적으로 면조명의 정밀한 표현에서 나타나는 다중적분을 몬테까를로방법으로 계산한다.

그림 14-72. 화소광선원추

화소표본화는 화소면우에 많은 광선들을 우연적으로 분산시키는 방법으로 진행한다. 그러나 광선위치들을 완전히 우연적으로 선택하는것은 광선들을 화소구역의 작은 부분구역에 밀집시키고 화소의 다른 부분들은 표본화되지 않은채로 남기는 결과를 초래할수 있다. 화소구역에서 빛분산의 더 좋은 근사는 정규부분화소격자에서의 순간요동화라고 하는 기술을 리용하여 얻을수 있다. 이것은 보통 처음에 화소구역(단위바른4각형)을 그림 14-73에 보여 준 16개의 부분구역들로 나누고 매개 부분구역에서 우연적인 순간요동위치를 발생시켜 얻는다. 여기서 δx와 δy에는 다같이 구간 (-0.5, 0.5)의 값들이 할당된다. 그러면 중심자리표가 (x, y)인 세포에서 광선위치는 순간요동위치 $(x+\delta x,\ y+\delta y)$로 선택된다.

옹근수코드 1~16은 매 16개 광선에 우연적으로 할당되며 다음에 설명되는바와 같이 표검색은 다른 파라메터(반사각, 시간 등)에 대한 값을 얻는데 리용된다. 매개 부분화소광선은 장면을 통하여 그 광선에 대한 세기몫을 결정하기 위하여 처리된다. 전체적인 화소세기를 만들기 위하여 16개의 광선세기를 평균한다. 부분화소들이 너무 많이 차이나면 화소는 앞으로 부분분할된다.

그림 14-73. 16개의 매 부분화소구역에 대하여 중심자리표로부터 순간요동된 광선위치를 리용하는 화소표본화

카메라렌즈효과를 모형화하기 위하여 초점거리가 f인 렌즈를 투영면앞에 설정하고 부분화소광선들을 렌즈구역우에서 분산시킨다. 화소당 16개의 광선을 가진다고 하면 렌즈구역을 16개의 부분구역으로 부분분할할수 있다. 매개 광선은 그의 할당된 코드에 따라 부분구역에 보내진다. 부분구역안에서 광선의 위치는 부분구역의 중심으로부터 순간요동된 위치로 설정된다. 다음에 광선은 순간요동된 부분구역위치로부터 렌즈의 초점을 통하여 장면에 투영된다. 광선에 대한 초점은 그림 14-74에 보여 준바와 같이 부분화소의 중심으로부터 렌즈중심을 통하는 선을 따라 렌즈로부터 거리 f에 있다. 초점면가까이의 물체들은 선명한 화상으로 투영된다. 초점면의 앞뒤에 있는 물체들은 희미해 진다. 초점이 맞지 않는 물체들을 더 잘 현시하기 위하여 부분화소광선들의 수를 증가시킨다.

면사귐점에서의 광선반사는 할당된 광선코드에 따라 거울반사방향 **R**에 대하여 분산된다(그림 14-75). **R**에 대한 최대분포는 16개 각부분구역으로 나뉘여 지며 매개 광선은 그의 옹근수코드에 대응하는 부분구역중심으로부터 순간요동된 위치에서 반사된다. 최대반사분산을 결정하기 위하여 퐁모형 $\cos^{n_s}\phi$를 리용할수 있다. 재질이 투명하면 굴절광선은 류사한 방식으로 투과방향 **T**에 대하여 분산된다.

그림 14-74. 초점거리가 f인 카메라렌즈우에서 부분화소광선들의 분산

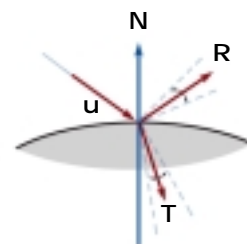



그림 14-75. 반사방향 **R**와 투과방향 **T**에 대한 부분화소광선들의 분산

확장광원은 그림 14-76에서 보여 주는바와 같이 많은 그림자광선들을 광원구역에 분산시키는 방법으로 처리한다. 광원은 부분구역들로 나뉘여 지며 그림자광선들에는 여러가지 부분구역에 대한 순간요동방향이 할당된다. 추가적으로 부분구역에는 그 부분구역안에서 광원의 세기와 물체면에 투영되는 부분구역의 크기에 따라 무게붙일수 있다. 더 많은 무게의 부분구역에는 더 많은 그림자광선들이 보내진다. 일부 그림자광선이 면과 광선사이의 불투명한 물체에 의하여 막히우면 그 면점에서 걸그늘이 발생된다. 그림 14-77은 광원으로부터 부분적으로 가리우는 면에서 속그늘과 걸그늘구역을 설명한다.

운동흐림은 광선들을 시간에 분산시켜 만든다. 총 프레임시간과 프레임시간부분분할은 장면에 대하여 요구되는 운동학에 따라 결정된다. 시간구간은 옹근수코드에 의하여 표식되며 매개 광선은 광선

코드에 대응하는 구간안에서 순간요동된 시간에 할당된다. 그러면 물체는 그 시간에 자기위치에로 움

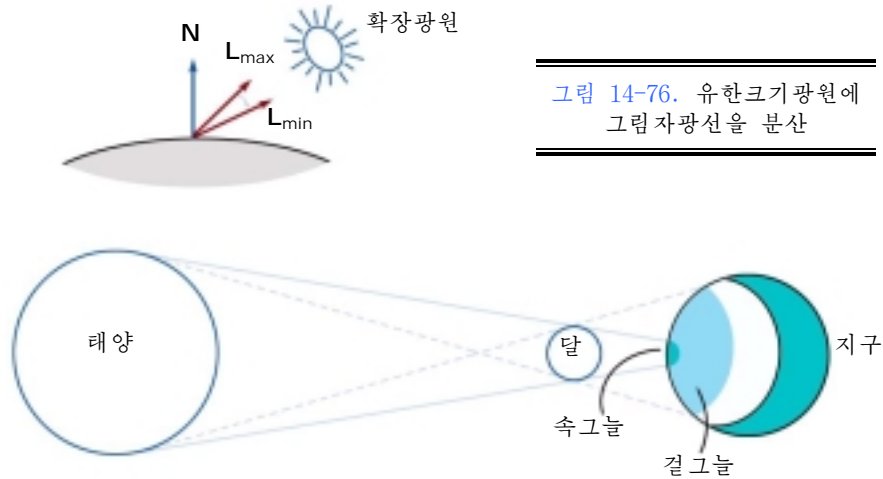



그림 14-76. 유한크기광원에 그림자광선을 분산

그림 14-77. 일식에 의하여 지구면에 만들어 지는 속그늘과 겉그늘구역

직여 지며 광선은 장면을 통하여 추적된다. 추가적인 광선들은 아주 희미해 진 물체들에 대하여 리용된다. 계산을 줄이기 위하여 초기광선사귐검사에 경계직6면체 또는 구를 리용할수 있다. 즉 경계물체를 운동의 요구에 따라 움직이고 사귐을 검사한다. 광선이 경계물체를 가로지르지 않으면 경계체적안의 개별적인 면들을 처리할 필요가 없다. 그림 14-78에서는 운동흐림현시된 장면을 보여 주었다. 이 화상은 4096×3550개의 화소와 화소당 16개의 광선에 의하여 분산광선추적방법으로 실감처리되였다. 운동흐림반사외에 그림자들은 당구탁을 조명하고 있는 방주위의 확장광원으로부터의 결과인 겉그늘구역들에 의해 현시된다.

그림 14-78. 분산광선추적에 의하여 실감처리된 1984 라는 이름의 장면(운동흐림과 겉그늘효과를 설명한다.)

분산광선추적방법에 의하여 실감처리된 물체들의 추가적인 실례를 그림 14-79와 14-80에 주었다. 그림 14-81에서는 분산광선추적에 의한 초점, 굴절, 경계허상제거효과들을 보여 주었다.

그림 14-79. 분산광선추적기술에 의하여 발생되는 반사 및 그림자효과를 보여 주는 윤기를 없앤 알루미니움 바퀴

그림 14-80. 분산광선추적방법으로 실감처리된 방장면

그림 14-81. 광선추적과 복사세기방법을 결합하여 가능한 초점, 경계허상제거, 조명효과를 보여 주는 장면(빛조명의 현실감 있는 물리모형은 유리잔의 그림자에서의 화선을 포함하여 굴절효과를 발생시키는데 리용되였다.)

7절. 복사세기조명모형

면에서의 확산반사는 에네르기보존법칙에 기초하여 면들사이의 복사에네르기이동을 고찰하는 방법으로 정밀하게 모형화할수 있다. 확산반사를 표현하는 이 방법을 일반적으로 **복사세기모형**(radiosity model)이라고 한다.

복사세기의 기본모형

이 방법에서는 장면안의 모든 면들사이의 복사에네르기의 호상작용을 고찰한다. 이를 위하여 장면안의 매개 면점을 떠나는 복사에네르기의 미분량 dB를 결정하고 면들사이에서의 에네르기이동량을 얻기 위하여 모든 면들에서 에네르기미분들을 합한다. 그림 14-82에서 dB는 면점에서 각 θ와 ϕ에 의하여 주어 지는 방향의 미소립체각 $d\omega$안에서 단위시간당 단위면적에서 나오는 보이는 복사에네르기이다. 그러므로 dB는 $J/(s\times m^2)$ 또는 W/m^2의 단위를 가진다.

방향 (θ, ϕ)에서 확산복사의 세기 또는 휘도 I는 단위시간동안에 단위투영면적에서 단위립체각당 복사에네르기이며 단위는 $W/(m^2\cdot sr)$이다.

$$I = \frac{dB}{d\omega \cos\phi} \qquad (14-68)$$

면을 리상적인 확산반사면이라고 하면 세기 I는 모든 보기방향에서 일정하게 설정할수 있다. 그러므로 $dB/d\omega$는 투영되는 면의 면적에 비례한다(그림14-83). 면점으로부터 단위시간동안에 복사되는 총 에네르기를 얻기 위하여 모든 방향에서 복사에네르기를 합한다. 즉 그림 14-84에서와 같이 면점에 중심이 있는 반구로부터 나오는 총 에네르기를 구한다.

$$B = \int_{\text{hemi}} dB \qquad (14-69)$$

완전한 확산반사면에 대하여 I는 상수이며 따라서 복사에네르기 B는

$$B = I \int_{\text{hemi}} \cos\phi \, d\omega \qquad (14-70)$$

와 같이 표현할수 있다. 또한 립체각의 미분요소 $d\omega$는

$$d\omega = \frac{dS}{r^2} = \sin\phi \, d\phi \, d\theta$$

와 같이 표현할수 있다(부록 1). 그러므로

$$\begin{aligned} B &= I \int_0^{2\pi} \int_0^{\pi/2} \cos\phi \sin\phi \, d\phi \, d\theta \\ &= I\pi \end{aligned} \qquad (14-71)$$

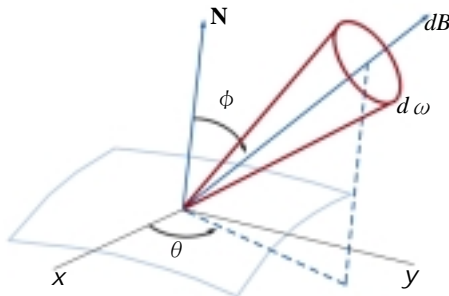



그림 14-82. 면점에서 (θ, ϕ)방향의 립체각 $d\omega$안에서 방출되는 보이는 복사에네르기

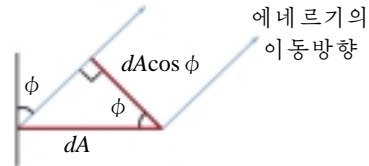



그림 14-83. 에네르기이동방향에 수직으로 투영되는 단위요소면의 면적은 $\cos\phi$와 같다.

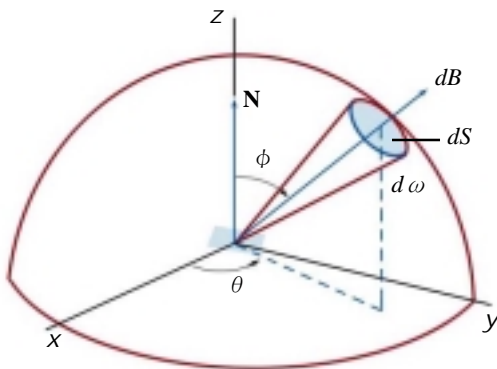



그림 14-84. 면점으로부터의 총 복사에네르기는 면점에 중심이 있는 반구의 모든 방향에서의 에네르기미분들의 합이다.

여러가지 면들로부터의 빛반사에 대한 모형은 면들의 《장벽》을 설정하여 만든다(그림 14-85). 장벽안의 매개 면은 반사기, 방출기(광원) 또는 반사기와 방출기의 결합이다. 복사세기파라메터 B_k를 단위시간동안에 면 k의 단위면적을 떠나는 에네르기로 지적한다. 입사에네르기파라메터 H_k는 담장안의 모든 면들로부터 단위시간동안에 면 k의 단위면적에 도착하는 에네르기들의 합이다. 즉

$$H_k = \sum_j B_j F_{jk} \tag{14-72}$$

여기서 파라메터 F_{jk}는 면 j와 k에 대한 형태결수(form factor)이다. 형태결수 F_{jk}는 면 j로부터 면 k에 닿는 복사에네르기의 비률이다.

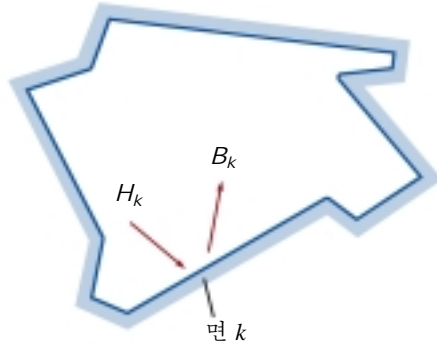



그림 14-85. 복사세기모형에서 면들의 장벽

담장안에 n개의 면이 있는 장면에서 면 k로부터의 복사에네르기는 복사세기방정식에 의하여 표현된다.

$$\begin{aligned} B_k &= E_k + \rho_k H_k \\ &= E_k + \rho_k \sum_{j=1}^{n} B_j F_{jk} \end{aligned} \tag{14-73}$$

면 k가 광원이 아니면 E_k=0이다. 그렇지 않으면 E_k는 단위시간동안에 면 k의 단위면적에서 방출되는 에네르기이다(W/m^2). 파라메터 ρ_k는 면 k의 반사률(입사빛에 대한 확산반사빛의 퍼센트)이다. 이 반사률은 경험적인 조명모형에서 리용되는 확산반사결수에 관계된다. 평면과 볼록면들은 자기자체를 볼수 없으며 그리하여 자체입사는 일어나지 않고 이 면들에 대한 형태결수 F_{kk}는 0이다.

담장안의 여러가지 면들에서의 조명효과를 얻자면 E_k, ρ_k, F_{jk}에 대한 배렬값들이 주어 진 n개의 면들에 대한 련립복사세기방정식을 풀어야 한다. 즉

$$(1-\rho_k F_{kk})B_k - \rho_k \sum_{j \neq k} B_j F_{jk} = E_k, \quad k = 1,2,3,\cdots,n \tag{14-74}$$

또는

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \cdot \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} \tag{14-75}$$

를 풀어야 한다. 다음에 복사세기값 B_k를 π로 나누어 세기값 I_k로 변환한다. 색장면에서는 ρ_k와 E_k의 색성분들로부터 복사세기의 개별적인 RGB성분(B_{kR}, B_{kG}, B_{kB})들을 계산할수 있다.

식 14-74를 풀기전에 형태결수 F_{jk}의 값들을 결정하여야 한다. 이를 위하여 면 j로부터 면 k에로의 에네르기이동을 고찰한다(그림14-86). 단위시간동안에 면소 dA_j로부터 면소 dA_k에 떨어 지는 복사

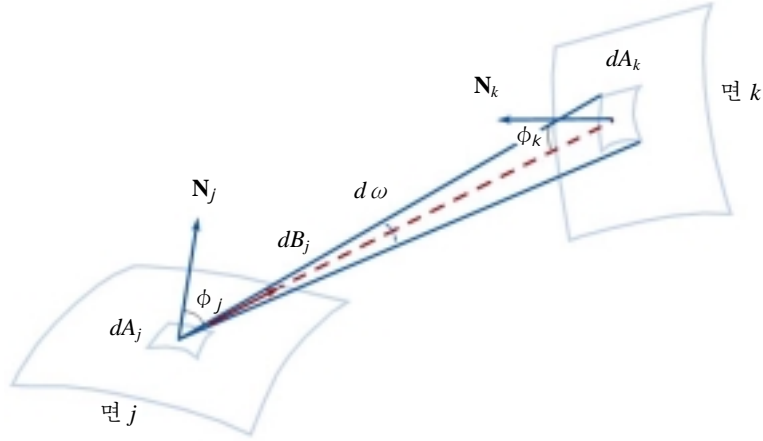



그림 14-86. 단위시간동안에 면적이 dA_j 인 면소로부터 면소 dA_k 로 이동된 에네르기 dB_j

에네르기는

$$dB_j dA_j = (I_j \cos\phi_j d\omega) dA_j \tag{14-76}$$

이다. 그런데 립체각 $d\omega$는 방향 dB_j에 수직인 면소 dA_k의 투영으로 쓸수 있다.

$$d\omega = \frac{dA}{r^2} = \frac{\cos\phi_k dA_k}{r^2} \tag{14-77}$$

그리하여 식 14-76은

$$dB_j dA_j = \frac{I_j \cos\phi_j \cos\phi_k dA_j dA_k}{r^2} \tag{14-78}$$

와 같이 표현할수 있다. 두면사이의 형태결수는 면적 dA_j로부터 나오는 에네르기가운데서 dA_k에 입사하는 에네르기의 비률이다.

$$\begin{aligned} F_{dA_j, dA_k} &= \frac{dA_k\text{에 입사하는 에네르기}}{dA_j\text{에서 나오는 전체 에네르기}} \\ &= \frac{I_j \cos\phi_j \cos\phi_k dA_j dA_k}{r^2} \cdot \frac{1}{B_j dA_j} \end{aligned} \tag{14-79}$$

또한 $B_j = \pi I_j$이며 따라서

$$F_{dA_j, dA_k} = \frac{\cos\phi_j \cos\phi_k dA_k}{\pi r^2} \tag{14-80}$$

이다. 면적 dA_j로부터 방출되는 에네르기가운데서 전체면 k에 입사하는 에네르기의 비률은

$$F_{dA_j, A_k} = \int_{\text{surf}_k} \frac{\cos\phi_j \cos\phi_k}{\pi r^2} dA_k \tag{14-81}$$

이다. 여기서 A_k는 면 k의 면적이다. 두 면사이의 형태결수는 앞식의 면적평균으로 정의할수 있다.

$$F_{jk} = \frac{1}{A_j} \int_{\text{surf}_j} \int_{\text{surf}_k} \frac{\cos\phi_j \cos\phi_k}{\pi r^2} dA_k dA_j \tag{14-82}$$

식14-82의 적분은 수값적분기술과 다음의 조건들을 규정하여 계산한다.

- $\sum_{k=1}^{n} F_{jk} = 1,$ 모든 k에 대하여(에네르기보존)
- $A_j F_{jk} = A_k F_{kj}$ (균등빛반사)
- $F_{jj} = 0,$ 모든 j에 대하여(평면 또는 볼록면쪼각들만을 가정한다.)

장면안의 매개 면은 많은 작은 다각형들로 부분분할될수 있으며 다각형의 면적이 더 작을수록 점점 더 현실감 있게 현시된다. 형태결수의 계산은 반구를 근사하는 반바른6면체를 리용하여 속도를 높일수 있다. 이때 구면을 선형면(평면)들의 모임으로 교체한다. 형태결수가 계산되면 레컨대 가우스소거법 또는 LU분해법을 리용하여 련립선형방정식 14-74를 풀수 있다. 또 다르게 B_j에 대한 근사값들로부터 가우스-자이델방법을 리용하여 련립선형방정식을 반복적으로 풀수 있다. 매개 반복에서 복사세기방정식

$$B_k = E_k + \rho_k \sum_{j=1}^{n} B_j F_{jk}$$

에서 앞서 얻어 진 복사세기값들을 리용하여 면쪼각 k에 대한 복사세기를 계산한다. 다음에 매개 걸음에서 장면을 현시할수 있으며 계산된 복사세기값에서 변화가 작을 때까지 매 반복에서 개선된 면실감처리된 장면을 본다.

복사세기의 점근법

비록 복사세기방법은 대단히 현실감 있게 면실감처리를 하지만 굉장히 큰 기억기가 요구되며 형태결수를 계산하는데는 상당한 처리시간이 필요하다. 점근법(progressive refinement)을 리용하여 계산속도를 높이고 매개 반복에서 기억기요구를 줄이기 위하여 반복복사세기알고리듬을 고쳐 만들수 있다.

복사세기방정식으로부터 두면쪼각사이의 복사세기묶은

$$B_j \text{ 로 인한 } B_k = \rho_k B_j F_{jk} \tag{14-83}$$

와 같이 계산된다. 반대로

$$B_k \text{ 로 인한 } B_j = \rho_j B_k F_{kj}, \quad \text{모든 } j \text{ 에 대하여} \tag{14-84}$$

이것은

$$B_k \text{ 로 인한 } B_j = \rho_j B_k F_{jk} \frac{A_j}{A_k}, \quad \text{모든 } j \text{ 에 대하여} \tag{14-85}$$

와 같이 다시 쓸수 있다. 이 관계는 복사세기계산에 대한 점근법의 기초이다. 단일면 쪼각 k를 리용하여 모든 형태결수 F_{jk}를 계산할수 있으며 그 쪼각으로부터 환경내의 다른 모든 면들에로 빛을 쏠수 있다. 그러므로 한번에 한개의 반바른6면체와 관련되는 형태결수를 계산하고 기억할수 있다. 다음에 이 값들을 버리고 다음번 반복에 대하여 다른 쪼각을 선택한다. 매 걸음에서 장면의 실감처리에 대한 근사를 현시한다.

초기에 모든 면쪼각들에 대하여 $B_k = E_k$로 설정한다. 다음에 제일 높은 복사세기값을 가지는 쪼각을 선택한다. 그것은 제일 밝은 광원이 될것이다. 그리고 다른 모든 쪼각들에 대하여 복사세기의 다음번 근사를 계산한다. 이 처리는 매 걸음에서 반복된다. 그리하여 광원들은 제일 높은 복사에네르기순서로 먼저 선택되며 다음에 다른 면쪼각들은 광원들로부터 받은 빛량에 기초하여 선택된다. 간단한 점근법에서의 걸음들을 다음의 알고리듬에 준다.

```
for each patch k
/*set up hemicube, calculate form factors F_jk*/

for each patch j{
          Δrad:=ρ_j B_k F_jk A_j /A_k;
          ΔB_j:=ΔB_j+ Δrad;
          B_j:=B_j+ Δrad;
}

ΔB_k=0;
```

그림 14-87. 복사세기의 점근모형에 의하여 John Wallace와 John Lin이 실감처리한 샤르뜨르대성당의 본당(Hewlett-Packard Starbase복사세기 및 광선추적쏘프트웨어를 리용. 복사세기의 형태결수들은 광선추적방법에 의하여 계산되였다.)

매 걸음에서 $\Delta B_k A_k$에 대한 제일 높은 값을 가지는 면쪼각이 발사쪼각으로 선택된다. 왜냐하면 복사세기가 단위면적당 복사에네르기의 크기이기때문이다. 그리고 모든 면쪼각들에 대하여 초기값을 $\Delta B_k = B_k = E_k$로 선택한다. 이 점근알고리듬은 장면을 통한 실제의 빛전파를 근사한다.

매 걸음에서 실감처리된 면들을 현시하는것은 어두운 장면으로부터 완전히 조명되는 장면에로 나아가는 보기렬을 만든다. 첫번째 걸음후 조명되는 면들은 광원과 선택된 그 광원에 보이는 비방출쪼각들뿐이다. 장면의 보다 쓸모 있는 초기보임상을 만들기 위하여 모든 쪼각들이 일부 조명을 가지도록 주변빛의 준위를 설정할수 있다. 반복의 매 단계에서 장면에 발사되는 복사에네르기량에 따라 주변빛을 줄인다.

그림 14-87에서는 복사세기의 점근모형에 의하여 실감처리된 장면을 보여 주었다. 여러가지 조명조건에 의한 복사세기실감처리된 장면들을 그림 14-88∼14-90에서 보여 주었다. 광선추적방법은 그림 14-81에서와 같이 대단히 현실감 있는 확산 및 거울면명암을 만들기 위하여 흔히 복사세기모형과 결합된다.

그림 14-88. 복사세기의 점근법에 의하여 실감처리된 구성주의의 박물관의 화상

그림 14-89. 복사세기의 접근법에 의하여 실감처리된 코넬종합대학의 공학리론쎈터건물의 계단탑의 모의

ㄱ)

ㄴ)

그림 14-90. 메트로폴리탄가극 《La Boheme》의 상연에서 빠리다락방에 대한 두가지 조명계획의 모의(ㄱ-낮의 모습, ㄴ-밤의 모습)

8절. 환경입히기

대역적인 반사를 모형화하는 또 다른 절차는 단일한 물체 또는 물체모임주위의 환경을 표현하는 세기값들의 배렬을 정의하는것이다. 이것은 거울 및 확산조명효과를 대역적으로 계산하는 광선추적이나 복사세기방법과는 달리 물체에 환경배렬을 보기방향에 따라 간단히 입힌다. 이 절차를 **환경입히기**(environment mapping)라고 하며 또한 **반사넘기기**(reflection mapping)라고도 한다. 투명효과도 환경배렬로 모형화할수 있다. 환경입히기는 때때로《거친 광선추적》방법이라고 한다. 왜냐하면 그것은 앞의 두개 절에서 설명한 정밀한 대역적인 조명실감처리기술에 대한 거친 근사이기때문이다.

환경배렬은 둘러 싸는 세계면우에서 정의된다. 환경배렬안의 정보에는 광원, 하늘, 다른 배경물체들에 대한 세기값이 속한다. 그림 14-91은 둘러싸는 세계를 구로 보여 주지만 바른6면체 또는 원기둥이 흔히 둘러싸는 세계로 리용된다.

물체의 면을 실감처리하기 위하여 화소구역을 면에 투영하고 매개 화소에 대한 면명암속성을 골라 내기 위하여 투영된 화소구역을 환경배렬에로 반사시킨다. 또한 물체가 투명하면 투영된 화소구역을 환경배렬에로 굴절시킬수 있다. 투영된 화소구역의 반사에 대한 환경입히기처리를 그림 14-92에 보여 주었다. 화소세기는 환경배렬의 사귐구역안의 세기값들을 평균하여 결정한다.

그림 14-91. 환경배렬을 포함하는 구모양의 둘러싸는 세계

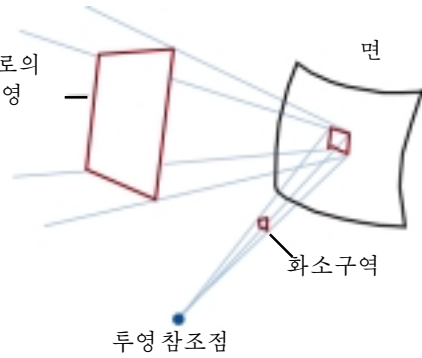



그림 14-92. 화소구역을 면에로 투영하고 다음에 그 구역을 환경배렬에로 반사

9절. 면세부의 추가

이제까지는 일반적으로 다각형 또는 스플라인과 같은 매끄러운 면을 현시하는 실감처리기술을 론의하였다. 그러나 대부분의 물체들은 매끄러운 즉 반반한 면을 가지지 않는다. 담장, 자갈길, 보풀이 이는 주단과 같은 물체들을 정밀하게 모형화하기 위하여 면의 결문양이 필요하다. 게다가 일부 면들에는 실감처리절차에서 고려되여야 할 무늬들이 있다. 꽃병의 면에는 색무늬가 있을수 있으며 물병의 면에는 가문의 표식무늬가 새겨 질수 있다. 정구경기장은 안쪽 측선과 바깥쪽 측선사이의 구역, 처넣기구역, 경기장 량끝의 한계선에 대한 표식들을 포함하며 4갈래 길의 도로는 나눔선과 함께 기름흘림, 다이야미끄럼과 같은 기타 표식들을 가진다. 그림 14-93에서는 여러가지 면 세부에 의하여 현시된 물체들을 보여 주었다.

ㄱ) ㄴ) ㄷ) ㄹ)

그림 14-93. 콤퓨터도형처리로 면세부들을 만든 장면들

다각형에 의한 면세부의 모형화

면에 세부를 추가하는 간단한 방법은 구조물과 무늬들을 다각형면쪼각에 의하여 모형화하는것이다. 큰 규모의 세부에 대하여 다각형모형화는 좋은 결과를 줄수 있다. 이러한 큰 규모의 세부의 몇가지 실례는 서양장기판의 눈, 대도로의 나눔선, 리놀리움바닥의 타일무늬, 매끄러운 키낮은 양털주단의 꽃무늬, 문의 판벽널, 경량화물자동차의 측면에 쓴 글자이다. 또한 불규칙적인 면은 방향이 우연적으로 지정되는 작은 다각형면쪼각들에 의하여 모형화할수 있는데 이때 면쪼각들은 너무 작지 않게 한다.

일반적으로 면무늬다각형들은 더 큰 면다각형에 중첩되며 어미면에 의하여 처리된다. 어미다각형만이 보이는 면알고리듬에 의하여 처리되지만 면세부다각형에 대한 조명파라메터들은 어미다각형에서 우선권을 가진다. 복잡한 또는 세밀한 면세부가 모형화될 때 다각형방법은 실천적으로 적합하지 않다. 실례로 건포도의 면구조는 다각형면쪼각으로 정밀하게 모형화하기가 힘들다.

겉문양입히기

면에 세부를 추가하는 일반적인 방법은 겉문양을 물체의 면에 입히는것이다. 겉문양은 직4각형배렬로 정의할수도 있고 면의 세기값을 수정하는 절차로도 정의할수 있다. 이 방법을 **겉문양입히기**(texture mapping) 또는 무늬입히기(pattern mapping)라고 한다.

보통 겉문양은 그림 14-94에 보여 준바와 같이 자리표값이 (s, t)로 지적되는 겉문양공간에서의 세기값의 직4각형격자로 정의된다. 장면안의 면우의 위치는 uv물체공간자리표로 지적되며 투영면에서의 화소위치는 xy직각자리표로 지적된다. 겉문양입히기는 두 방법중 하나로 수행할수 있다. 겉문양을 물체면에 넘기고 그다음 투영면에로 넘길수도 있으며 또는 화소구역을 물체면에 넘기고 다시 겉문양공간에로 넘길수도 있다. 때때로 겉문양을 화소자리표에로 넘기는것을 겉문양주사(texture scanning)라고 하며 한편 화소자리표를 겉문양공간에로 넘기는것은 화소급주사(pixel-order scanning), 역주사(inverse scanning) 또는 화상급주사(image-order scanning)라고 한다.

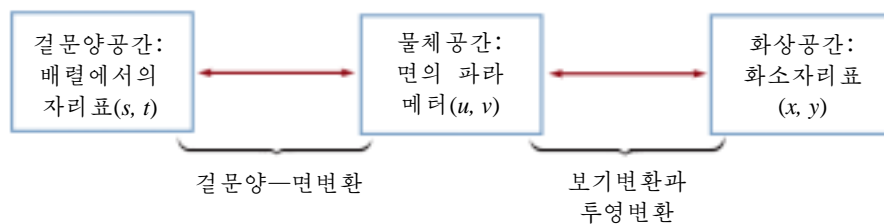



그림 14-94. 겉문양공간, 물체공간, 화상공간에 대한 자리표계

계산을 간단히 하기 위하여 겉문양공간으로부터 물체면에로의 넘기기는 대체로 보조변수선형함수

$$u = f_u(s,t) = a_u s + b_u t + c_u$$
$$v = f_v(s,t) = a_v s + b_v t + c_v \qquad (14-86)$$

에 의하여 지적된다. 물체-화상공간넘기기는 보기 및 투영변환들을 결합하는 방법으로 진행한다. 겉문양공간으로부터 화소공간에로의 넘기기는 선택된 겉문양쪼각이 보통 화소경계와 정합되지 않는 결함을 가지기때문에 화소가 차지하는 부분의 면적을 계산하여야 한다. 그러므로 화소공간으로부터 겉문양공간에로의 넘기기가 가장 일반적으로 리용되는 겉문양입히기방법이다(그림 14-95). 이것은 화소부분분할계산을 피하며 경계허상제거(려파)절차가 쉽게 적용될수 있게 한다. 효과적인 경계허상제거절차는 그림 14-96에 보여 준바와 같이 린접화소들의 중심을 포함하는 약간 더 큰 화소구역을 투영

하고 결문양에서의 세기값들에 무게를 다는데 각추함수를 적용하는것이다. 그러나 화상공간으로부터 결문양공간에로의 넘기기는 역보기투영변환 $\mathbf{M}^{-1}_{VP}$과 역결문양변환 $\mathbf{M}^{-1}_{T}$의 계산을 요구한다. 다음의 실례에서 정의된 무늬를 원기둥면에 넘겨 이 방법을 설명한다.

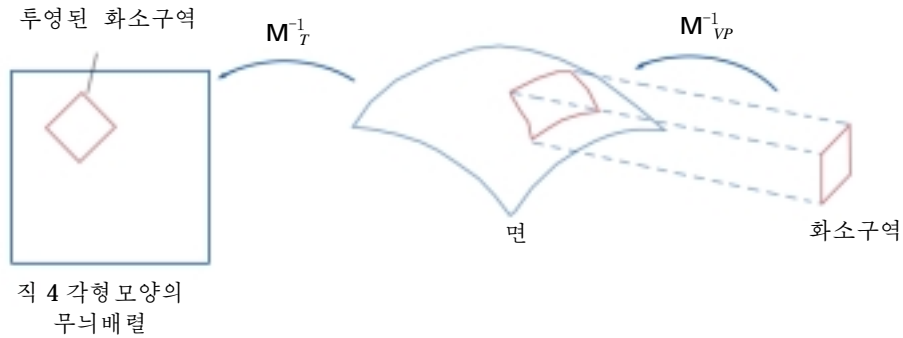



그림 14-95. 화소구역을 결문양공간에로 투영하는것에 의한 결문양입히기

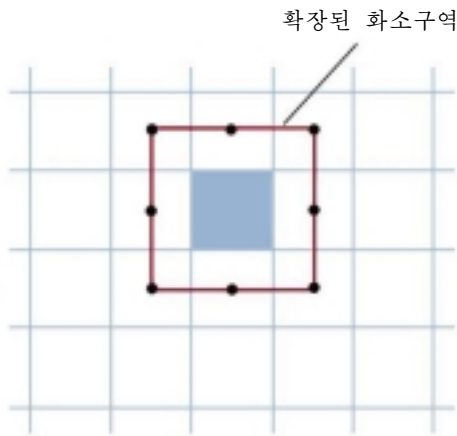



그림 14-96. 린접화소들의 중심을 포함하는 확장된 화소구역

실례 14-1. 결문양입히기

결문양입히기에서의 걸음들을 설명하기 위하여 그림 14-97에 보여 준 무늬를 원기둥면에 넘기는것을 고찰한다. 면의 파라메터들은

$$u=\theta, \qquad v=z$$

여기서

$$0\le\theta\le\pi/2, \qquad 0\le z\le 1$$

이다. 그리고 직각자리표계에서 면에 대한 보조변수표현은

$$x=r\cos u, \qquad y=r\sin u, \qquad z=v$$

이다. 배렬무늬는 무늬의 원점을 면의 제일 왼쪽 아래구석에 넘기는 다음의 선형변환에 의하여 면에 넘길수 있다.

$$u=s\pi/2, \qquad v=t$$

다음에 보기위치를 선택하고 화소자리표로부터 원기둥면우의 직각자리표에로의 역보기변환을 수행한다. 직각자리표는 다음에 변환

$$u=\tan^{-1}(y/x), \quad v=z$$

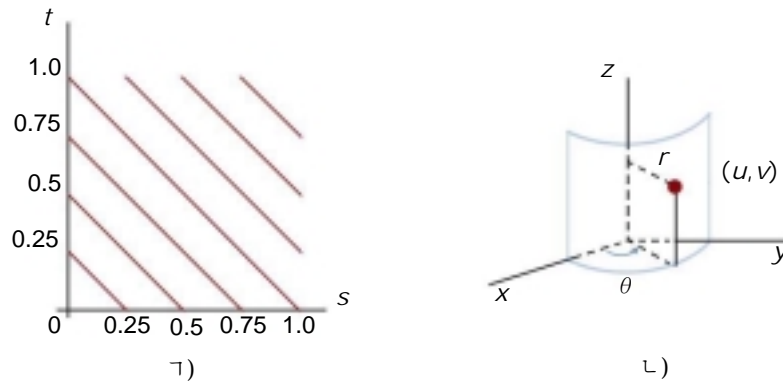



그림 14-97. 단위바른4각형에서 정의된 걸문양(ㄱ)을 원기둥면(ㄴ)에로 넘기기

에 의하여 면의 파라메터들로 넘겨 진다. 그리고 투영된 화소위치는 역변환

$$s=2u/\pi, \qquad t=v$$

에 의하여 걸문양공간에로 넘겨 진다. 매개 투영된 화소로 덮여 지는 무늬배렬에서의 세기값들은 화소의 세기를 얻기 위하여 평균된다.

수속적인 걸문양입히기

면에 걸문양을 추가하는 다른 방법은 장면안의 물체에 가해 질 색의 변화에 대한 수속적인 정의를 리용하는것이다. 이 방법에서는 2차원걸문양을 물체면에 넘기는 변환계산을 하지 않아도 된다.

값들이 3차원공간구역에서 주어 졌을 때 물체의 색변화를 립체걸문양(solid texture)이라고 한다. 공간구역안의 모든 점들에 대한 걸문양값을 기억하는것은 보통 불가능하기때문에 걸문양공간의 값들은 수속적인 방법을 리용하여 물체면에 넘겨 진다. 다른 수속적인 방법들은 2차원면우에서 걸문양값을 설정하는데 리용될수 있다. 립체걸문양입히기는 벽돌과 같은 3차원물체의 자름면이 바깥면과 같은 걸문양짜임새로 실감처리될수 있게 한다.

수속적인 걸문양입히기의 실례로 나무의 결이나 대리석무늬는 3차원공간에서 정의되는 조화함수(시누스곡선)를 리용하여 만들수 있다. 나무나 대리석의 걸문양짜임새에서의 우연적인 변화는 잡음함수를 조화변환에 덧붙여 얻을수 있다. 그림 14-98에서는 나무의 결과 기타 면무늬들을 얻는데 립체걸문양을 리용하여 현시된 장면을 보여 주었다. 그림 14-99의 장면은 세공돌, 윤기나는 금, 바나나잎사귀와 같은 재질들의 수속적인 표현을 리용하여 실감처리되였다.

그림 14-98. 면들의 특징을 립체걸문양방법을 리용하여 만든 장면

곰보만들기

비록 걸문양입히기는 세밀한 면세부를 추가하는데 리용될수 있지만 귤, 양딸기, 건포도와 같은 물체들에서 나타나는 면의 울퉁불퉁함을 모형화하는데는 좋은 방법이 아니다. 걸문양에서의 세부조명은 보통 장면에서의 조명방향에 일치하지 않는다. 면에 곰보를

만드는 더 좋은 방법은 면의 법선에 섭동함수를 적용

그림 14-99. 보석면, 2차곡면, 쌍3차곡면쪼각에 대하여 다각형면쪼각을 리용하는 RenderMan에 의하여 모형화되고 VGShader에 의하여 실감처리된 장면(걸문양입히기외에 수속적인 방법들은 안개가 짙은 원시림의 대기와 숲을 뒤덮은 얼룩진 조명효과를 만드는데 리용되였다.)

하고 다음에 조명모형계산에서 섭동된 법선을 리용하는것이다. 이 기술을 **곰보만들기**(bump mapping)라고 한다.

$\mathbf{P}(u, v)$가 보조변수곡면우의 위치를 표현하면 그 점에서의 면의 법선은 계산식

$$\mathbf{N} = \mathbf{P}_u \times \mathbf{P}_v \tag{14-87}$$

에 의하여 얻을수 있다. 여기서 $\mathbf{P}_u$와 $\mathbf{P}_v$는 파라메터 u와 v에 관한 $\mathbf{P}$의 편도함수들이다. 섭동된 법선을 얻기 위하여 면의 위치벡토르를 곰보함수라고 하는 작은 섭동함수를 추가하여 수정한다.

$$\mathbf{P}'(u,v) = \mathbf{P}(u,v) + b(u,v)\mathbf{n} \tag{14-88}$$

이것은 면의 단위법선 $\mathbf{n} = \mathbf{N} / |\mathbf{N}|$의 방향에서 면에 곰보를 추가한다. 그러면 면의 섭동된 법선은

$$\mathbf{N}' = \mathbf{P}_u' \times \mathbf{P}_v' \tag{14-89}$$

와 같이 얻어 진다. 섭동된 위치벡토르의 u에 관한 편도함수는

$$\begin{aligned}\mathbf{P}_u' &= \frac{\partial}{\partial u}(\mathbf{P} + b\mathbf{n}) \\ &= \mathbf{P}_u + b_u\mathbf{n} + b\mathbf{n}_u\end{aligned} \tag{14-90}$$

와 같이 계산된다. 곰보함수 b가 작다고 하면 마지막항을 무시하고

$$\mathbf{P}_u' \approx \mathbf{P}_u + b_u\mathbf{n} \tag{14-91}$$

와 같이 쓸수 있다. 류사하게

$$\mathbf{P}_v' \approx \mathbf{P}_v + b_v\mathbf{n} \tag{14-92}$$

그리고 면의 섭동된 법선은

$$\mathbf{N}' = \mathbf{P}_u \times \mathbf{P}_v + b_v(\mathbf{P}_u \times \mathbf{n}) + b_u(\mathbf{n} \times \mathbf{P}_v) + b_u b_v(\mathbf{n} \times \mathbf{n})$$

이다. 그런데 $\mathbf{n} \times \mathbf{n} = 0$이며 따라서

$$\mathbf{N}' = \mathbf{N} + b_v(\mathbf{P}_u \times \mathbf{n}) + b_u(\mathbf{n} \times \mathbf{P}_v) \tag{14-93}$$

마지막걸음은 조명모형계산에 리용하기 위하여 $\mathbf{N}'$를 정규화하는것이다.

곰보함수 $b(u, v)$를 지적할수 있는 여러가지 방법들이 있다. 실제로 해석적인 식을 정의할수 있지만 곰보값은 보통 표검색에 의하여 얻어 진다. 곰보표에서 b의 값은 선형보간과 증분계산에 의하여

빨리 얻을수 있다. 편도함수 b_u와 b_v는 유한차에 의하여 근사된다. 곰보표는 우연무늬, 규칙격자무늬 또는 특성형태에 의하여 설정될수 있다. 우연무늬는 건포도와 같은 불규칙적인 면을 모형화하는데 쓸모 있으며 한편 반복무늬는 귤의 면을 모형화하는데 리용될수 있다. 경계허상을 제거하기 위하여 화소구역을 부분분할하고 계산된 보조화소세기들을 평균한다.

그림 14-100에서는 곰보만들기에 의하여 실감처리된 면들의 실례를 보여 주었다. 면실감처리방법들이 결합된 실례는 그림 14-101에 주었다. 영화 《젊은 샬로크홈스》에서 색유리무사의 갑옷은 곰보만들기, 환경입히기, 결문양입히기를 결합하여 실감처리하였다. 주변의 환경배렬은 배경조명반사와 면의 울퉁불퉁함을 만들기 위하여 곰보배렬과 결합되였다. 다음에 그림 14-101에 보여 준 전반적인 효과를 만들기 위하여 추가적인 색과 면조명, 곰보, 먼지의 국부조명광선, 이은 자리와 리베트에 대한 얼룩이 추가되였다.

ㄱ)

ㄴ)

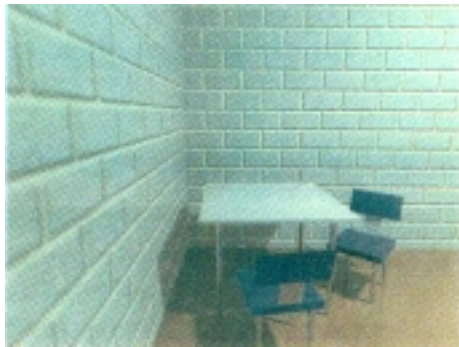



그림 14-100. 곰보만들기에 의하여 실감처리된 면의 울퉁불퉁한 특성

그림 14-101. 영화 《젊은 샬로크홈스》에서의 색유리무사(갑옷면을 실감처리하는데 곰보만들기, 환경입히기, 결문양입히기를 함께 리용하였다.)

주름만들기

이 기술은 곰보만들기의 확장이다. 주름만들기(frame mapping)에서는 면의 법선 **N**과 **N**에 접속되는 국부자리표계(그림 14-102)를 다같이 섭동시킨다. 국부자리표계는 면의 접선벡토르 **T**와 배법선벡토르 **B**=**T**×**N**에 의하여 정의된다.

주름만들기는 이방성면을 모형화하는데 리용된다. **N**방향에서의 곰보섭동외에 **T**를 면의 결을 따라 방위선정하고 방향적인 섭동을 적용한다. 이 방법으로 나무의 결무늬, 천의 짜임새, 대리석 또는 그와 류사한 재질들에서의 줄무늬를 모형화할수 있다. 곰보 및 방향섭동은 다같이 표검색에 의하여 얻을수 있다.

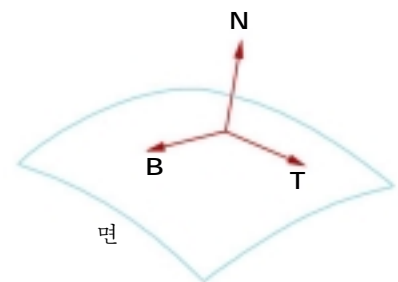



그림 14-102. 면우의 점에서의 국부자리표계

요약

일반적으로 물체는 장면안에서 광원과 기타 물체들의 반사면들로부터의 복사에네르기에 의하여 조명된다. 광원은 점광원이나 분산광원(확장광원)으로 모형화될수 있다. 물체는 불투명하거나 투명할수 있다. 그리고 조명효과는 반사와 굴절에 대하여 다같이 확산성분과 거울성분으로 표현할수 있다.

경험적인 조명모형(점광원)은 람버트의 코시누스법칙에 의하여 확산반사를 표현하고 퐁모형에 의하여 거울반사를 표현하는데 리용될수 있다. 일반적인 배경(주변)조명은 매면에 대하여 고정된 세기준위와 반사결수에 의해 모형화될수 있다. 이 기본모형에서 투명결수를 리용하여 면의 세기들을 결합함으로써 투명효과를 근사시킬수 있다. 투명한 재질속에서 빛경로의 정확한 기하모형화는 스넬의 법칙을 리용하여 굴절각을 계산하여 얻는다. 색은 RGB값들의 조를 세기와 면의 반사결수에 할당하여 모형에 병합한다. 또한 분산광원, 촬영장조명효과, 세기감쇠를 병합하도록 기본모형을 확장할수 있다.

조명모형에 의하여 계산된 세기값은 사용하는 현시체계에서 사용가능한 세기준위로 넘겨 져야 한다. 세기의 로그척도는 등감각밝기를 가지는 세기준위들의 모임을 제공하는데 리용된다. 감마보정은 현시장치들의 비선형성을 보정하기 위하여 세기값들에 적용된다. 2값현시장치들에서 세기값들의 범위를 모의하기 위하여 중간명암무늬와 한정색표시기술을 리용할수 있다. 중간명암근사는 화소당 두개이상의 세기들을 현시할수 있는 체계들에서 세기항목들의 개수를 증가시키는데도 리용할수 있다. 장면에서 표시될 점들의 개수가 현시장치우의 화소들의 개수와 같을 때 세기들의 범위를 모의하는데 순서디더, 오차확산, 점확산방법들이 리용된다.

면실감처리는 기본조명모형을 장면안의 물체들에 적용하여 진행할수 있다. 상수세기명암처리, 그로우명암처리 또는 퐁명암처리를 리용하여 조명모형을 적용한다. 상수세기명암처리는 보기위치와 광원의 위치가 장면안의 물체들로부터 멀 때 다면체와 곡면다각형그물에 대하여 정확하다. 그로우명암처리는 다각형의 정점들에서의 세기값들을 계산하고 다각형쪼각들을 가로질러서 이 세기값들을 보간하여 곡면에서의 빛반사를 근사시킨다. 보다 정확하지만 더 느린 면실감처리절차는 퐁명암처리이다. 그것은 다각형쪼각우에서 다각형의 정점들에 대한 평균법선벡토르들을 보간한다. 그러면 면의 세기들은 보간된 법선벡토르들을 리용하여 계산된다. 고속퐁명암처리는 테일러합렬근사를 리용하여 계산속도를 높이는데 리용될수 있다.

광선추적은 대역적인 거울반사와 투과효과들을 얻기 위한 정확한 방법을 제공한다. 화소광선은 한 물체에서 다른 물체에로 튀여 나며 한편 세기몫들을 축적하면서 장면을 통하여 추적된다. 광선추적나무는 매 화소에 대하여 구성되며 세기값들은 나무의 말단마디로부터 거꾸로 뿌리까지 결합된다. 광선추적에서의 물체사귐점계산은 전체 공간의 부분구역안에서만 광선-물체사귐을 시험하는 공간부분분할방법들에 의하여 줄여 질수 있다. 분산된(또는 분산) 광선추적은 화소당 다중광선을 추적하며 방향과 시간과 같은 여러가지 광선파라메터들우에서 광선들을 우연적으로 분산시킨다. 이것은 면의 광택과 반투명,유한카메라시야조절장치,분산광원,그림자효과 그리고 운동흐림을 모형화하기 위한 정확한 방법을 제공한다.

복사세기방법은 장면안의 여러가지 곡면쪼각들사이에서의 복사에네르기이동을 계산하는 방법으로 확산반사효과의 정확한 모형화를 제공한다. 점근법은 한번에 하나의 곡면쪼각으로부터의 에네르기이동을 고찰함으로써 복사세기의 계산속도를 높이는데 리용된다. 사진같은 장면들은 광선추적과 복사세기를 결합하여 만들어 진다.

대역조명효과를 근사하기 위한 빠른 방법은 환경입히기이다. 환경입히기는 장면에 대한 배경세기정보를 기억하는데 리용된다. 이 배렬은 다음에 지적된 보기방향에 기초하여 장면안의 물체들에 넘겨 진다.

면세부는 다각형쪼각, 걸문양입히기, 곰보만들기, 주름만들기를 리용하여 물체들에 추가될수 있다. 여러가지 종류의 설계들을 제공하기 위하여 작은 다각형쪼각들은 더 큰 면들우에 중첩될수 있다. 또 다르게 걸문양은 2차원배렬로 정의될수 있으며 물체면들에 넘겨 질수 있다. 곰보만들기는 면의 법선들을 섭동시키는데 곰보함수를 적용하여 면의 불규칙성을 모형화하는 방법이다. 주름만들기는 수직변화뿐아니라 수평면변화를 하게 하는 곰보만들기의 확장이다.

참고문헌

에네르기전파, 이동방정식, 실감처리, 빛과 색의 감각에 대한 일반적인 설명은 Glassner(1994)에 주었다. 여러가지 면실감처리기법들에 대한 알고리듬들은 Glassner(1990), Arvo(1991), Kirk(1992)에 제출되였다. 순서디더, 오차확산, 점확산의 그이상의 설명은 Knuth(1987)을 보시오. 광선추적방법에서의 추가적인 정보는 Quek와 Hearn(1988), Glassner(1989), Shirley(1990), Koh와 Hearn(1992)에 주었다. 복사세기방법은 Goral 등(1984), Cohen과 Greenberg(1985), Cohen 등(1988), Wallace, Elmquist, Haines(1989), Chen등(1991), Dorsey, Sillion, Greenberg(1991), He등(1992), Sillion등(1991), Schoeneman 등(1993), Lischinski, Tampieri, Greenberg(1993)에서 설명하였다.

련습문제

14-1. 지적된 다면체의 면들에 대하여 단일한 점광원과 면의 상수세기명암처리를 리용할 때 기본조명모형의 식 14-4를 실현하는 루틴을 쓰시오. 물체는 매 다각형면에 대한 면의 법선들을 비롯한 다각형표들의 모임으로 표현한다. 추가적인 입력파라메터들에는 주변세기, 광원의 세기, 면의 반사결수들이 있다. 모든 자리표정보는 보기자리표계에서 직접 지적될수 있다.

14-2. 다각형면들을 실감처리하기 위하여 그로우명암처리를 써서 련습문제 14-1의 루틴을 수정하시오.

14-3. 다각형면들을 실감처리하기 위하여 퐁명암처리를 써서 련습문제 14-1의 루틴을 수정하시오.

14-4. 지적된 다면체의 면들에 대하여 단일한 점광원과 그로우명암처리를 리용할 때 기본조명모형의 식 14-9를 실현하는 루틴을 쓰시오. 물체표현은 매 다각형면에 대한 면의 법선들을 포함하여 다각형표들의 모임으로 주어 질수 있다. 추가적인 입력에는 주변세기, 광원의 세기, 면의 반사결수, 거울반사파라메터에 대한 값들이 있다. 모든 자리표정보는 보기자리표계에서 직접 지적될수 있다.

14-5. 다각형면들을 실감처리하기 위하여 퐁명암처리를 써서 련습문제 14-4의 루틴을 수정하시오.

14-6. 선형세기감쇠함수를 포함하도록 련습문제 14-4의 루틴을 수정하시오.

14-7. 다각형면들을 실감처리하기 위하여 퐁명암처리와 선형세기감쇠함수를 써서 련습문제 14-4의 루틴을 수정하시오.

14-8. 장면안에서 임의의 개수의 다면체와 광원이 지적되였을 때 식 14-13을 실현하기 위하여 련습문제 14-4의 루틴을 수정하시오.

14-9. 장면안에서 임의의 개수의 다면체와 광원이 지적되였을 때 식 14-14를 실현하기 위하여 련습문제 14-4의 루틴을 수정하시오.

14-10. 장면안에서 임의의 개수의 다면체와 광원이 지적되였을 때 식 14-15를 실현하기 위하

여 련습문제 14-4의 루틴을 수정하시오.

14-11. 장면에서 임의의 개수의 광원과 다면체(불투명 또는 투명)가 지적되였을 때 식14-15와 14-19를 실현하기 위하여 련습문제 14-4의 루틴을 수정하시오.

14-12. $(\mathbf{V}\cdot\mathbf{R})^{n_s}$에 의하여 모형화된 거울반사와 $(\mathbf{N}\cdot\mathbf{H})^{n_s}$에 의하여 모형화된 거울반사를 비교하고 차이점을 설명하시오.

14-13. 그림 14-18에서 모든 벡토르들이 동일한 면에 놓일 때 $2\alpha=\phi$이지만 일반적으로 $2\alpha\neq\phi$라는것을 증명하시오.

14-14. 불투명한 면들을 가지는 다면체들의 모임을 현시하기 위하여 서로 다른 보이는 면검출방법들이 어떻게 세기모형과 결합될수 있는가를 설명하시오.

14-15. 투명한 물체들을 처리하기 위하여 여러가지 보이는 면검출방법들이 어떻게 수정될수 있는가를 설명하시오. 투명한 면들을 처리할수 있는 보이는 면검출방법이 있는가?

14-16. 보이는 면검출방법들중의 하나를 리용하여 먼 거리의 점광원에 의하여 조명되는 장면에서 그림자구역들을 식별하게 될 알고리듬을 설정하시오.

14-17. 매개 화소가 매개의 서로 다른 세기들에 의하여 현시될수 있는 n×n화소격자를 리용하면 중간명암근사에 의하여 얼마나 많은 세기준위들이 현시될수 있는가?

14-18. 2값 RGB체계의 3×3화소격자에서 중간명암근사를 리용하면 얼마나 많은 색결합들을 만들수 있는가?

14-19. 3×3화소격자와 화소당 두개의 세기준위(0과 1)를 가질 때 중간명암근사를 리용하여 면세기변화들의 주어 진 모임을 현시하는 루틴을 쓰시오.

14-20. 식 14-34의 재귀관계를 리용하여 순서데이자행렬들을 만드는 루틴을 쓰시오.

14-21. 순서디더방법을 리용하여 세기값들의 주어 진 배렬을 현시하는 절차를 쓰시오.

14-22. 세기값들의 주어 진 m×n배렬에 대하여 오차확산알고리듬을 실현하는 절차를 쓰시오.

14-23. 서양장기판의 바닥눈우에 하나의 구가 떠 있는 장면에 대하여 광선추적의 기본알고리듬을 실현하는 프로그람을 쓰시오. 장면은 보기위치에 있는 단일한 점광원에 의하여 조명될것이다.

14-24. 구와 다각형면들의 임의로 지적된 배렬이 있으며 점광원들의 주어 진 모임에 의하여 조명되는 장면에 대하여 광선추적의 기본알고리듬을 실현하는 프로그람을 쓰시오.

14-25. 점광원들의 주어 진 모임에 의하여 조명되는 구와 다각형면들의 임의로 지적된 배렬에 대하여 공간부분분할방법을 써서 광선추적의 기본알고리듬을 실현하는 프로그람을 쓰시오.

14-26. 분산광선추적의 다음의 특징들을 실현하는 프로그람을 쓰시오. 화소당 16개의 순간요동광선을 가지는 화소표본화, 분산된 반사방향, 분산된 굴절방향, 확장광원

14-27. 분산광선추적을 리용하여 움직이는 물체의 운동흐림을 모형화하기 위한 알고리듬을 설정하시오.

14-28. 바른6면체의 하나의 내부면이 광원일 때 바른6면체의 내부면들을 실감처리하기 위한 복사세기의 기본알고리듬을 실현하시오.

14-29. 복사세기의 점근법을 실현하기 위한 알고리듬을 창안하시오.

14-30. 환경배렬를 구면에로 변환하는 루틴을 쓰시오.

14-31. 구면과 다면체에 대하여 걸문양입히기를 실현하는 프로그람을 쓰시오.

14-32. 구면이 주어 졌을 때 귤의 울퉁불퉁한 면을 만드는 곰보만들기절차를 쓰시오.

14-33. 임의로 지적된 곰보함수에 대하여 면의 법선변화를 만드는 곰보만들기루틴을 쓰시오.

15장. 색모형과 색응용

이때까지의 색에 대한 설명은 붉은색, 풀색, 푸른색빛을 결합하여 색을 현시하는 장치들에 집중하였다. 이 모형은 현시장치에서 색이 어떻게 표현되는가를 리해하는데 도움이 되지만 도형처리응용에서는 여러가지 다른 색모형들도 또한 쓸모 있다. 일부 모형들은 인쇄기와 작도기에서의 색출력을 표현하는데 리용되며 다른 모형들은 사용자에게 보다 직관적인 색선택대면부를 제공한다.

색모형(color model)은 어떤 구체적인 문맥에서 색의 특성이나 작용을 설명하는 방법이다. 모든 색상황을 다 설명할수 있는 단일한 색모형은 없으며 그러므로 서로 다르게 감각되는 색의 특성들을 쉽게 표현하기 위하여 여러가지 모형들을 리용한다.

1절. 빛의 특성

우리가 감각하는 빛 또는 색은 전자기스벡트르안의 좁은 주파수대역이다. 전자기스펙트르안에는 무선파, 마이크로파, 적외선파, X선이라고 부르는 몇가지 다른 주파수대역들이 있다. 그림 15-1은 몇가지 전자기대역들의 근사주파수범위를 보여 준다.

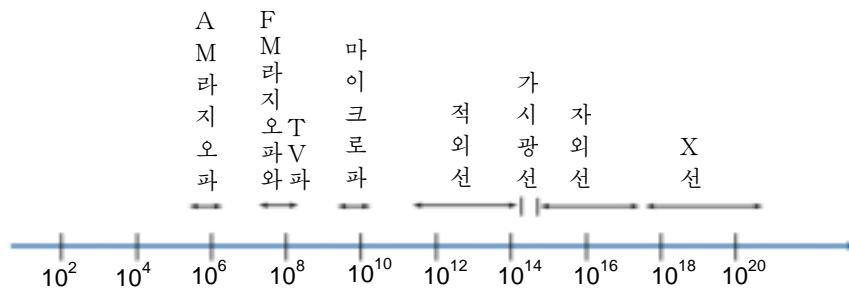



그림 15-1. 전자기스펙트르

가시대역안의 매개 주파수값은 서로 다른 색에 대응한다. 가시대역의 제일 낮은 주파수는 붉은색(4.3×10^{14}Hz)이며 제일 높은 주파수는 보라색(7.5×10^{14}Hz)이다. 스펙트르색은 낮은 주파수의 붉은색으로부터 감색, 누런색을 거쳐 높은 주파수의 풀색, 푸른색, 보라색까지의 범위에 있다.

빛은 전자기파이기때문에 색은 주파수 f나 파장 λ에 의하여 표현할수 있다. 그림 15-2에서는 전기적진동들이 한 평면에 있도록 편광된 단색전자기파의 진동을 보여 주었다. 단색파의 파장과 주파수는 서로 역비례하며 비례상수는 빛속도 c이다.

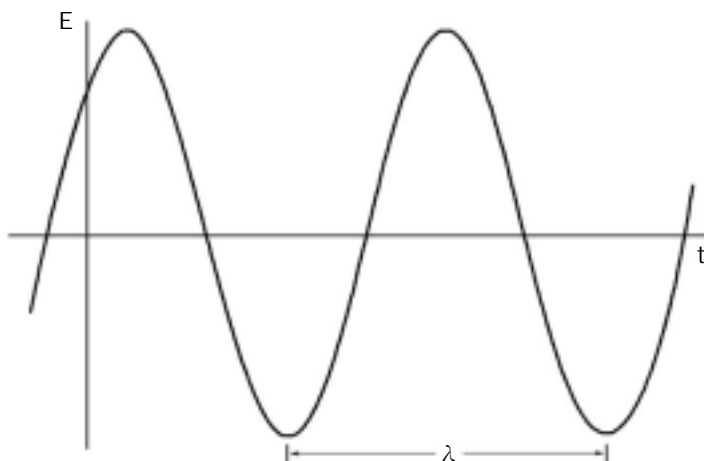



그림 15-2. 평면편광된 전자기파의 한개 전기주파수성분에 대한 시간에 따르는 변화

$$c = \lambda f \qquad (15-1)$$

주파수는 모든 재질들에 대하여 일정하지만 빛속도와 파장은 재질에 따라 다르다. 진공에서 $c=3\times10^{10}$cm/s이다. 빛의 파장은 대단히 작으며 그리하여 스펙트르색을 지적하는 길이의 단위는 보통 옹그스트롱(1Å=10^{-8}cm)이나 나노메터(1nm=10^{-7}cm)이다. 나노메터의 동의어는 미리마이크로메터이다. 스펙트르의 한 끝에서 붉은색빛의 파장은 약 700nm이며 스펙트르의 다른 끝에서 보라색빛의 파장은 약 400nm이다. 파장의 단위는 주파수의 단위보다 다루기가 좀더 편리하기때문에 스펙트르색은 일반적으로 파장에 의하여 지적한다.

태양이나 백열등과 같은 광원들은 백색빛을 만들기 위하여 가시범위안의 모든 주파수들을 방출한다. 백색빛이 물체에 입사할 때 일부 주파수들은 반사되고 일부는 물체에 의하여 흡수된다. 반사빛에 있는 주파수들의 결합은 우리가 감각하는 물체의 색을 결정한다. 반사빛에서 낮은 주파수들이 우세하면 물체는 붉은색으로 표현된다. 이 경우에 감각되는 빛은 스펙트르의 붉은색부에서 기본주파수(또는 주파장)를 가진다고 한다. 기본주파수는 빛의 색채(hue) 또는 간단히 색(color)이라고도 한다.

빛의 여러가지 특성을 표현하는데는 주파수외에 다른 특성량들이 필요하다. 광원에서 나오는 빛을 볼 때 사람의 눈은 색(또는 기본주파수)과 기타 두가지 기본감각에 반응한다. 이것들중 하나는 밝기라고 부르며 그것은 빛의 감각적인 세기이다. 빛의 감각적인 세기는 단위시간동안에 단위립체각에서 광원의 단위투영면적으로부터 방출되는 빛에네르기이며 이것은 광원의 휘도에 관계된다. 두번째 감각특성량은 빛의 순도(purity) 또는 포화도(saturation)이다. 순도는 빛의 색이 연하고 진한 정도를 나타낸다. 파스텔색과 연한 색은 덜 순수하다고 말한다. 이 세가지 특성량 즉 기본주파수, 밝기, 순도는 일반적으로 빛의 서로 다른 특성을 표현하는데 리용된다. 용어 색도(chromaticity)는 색의 특성을 표현하는 두가지 특성량 즉 순도와 기본주파수를 집합적으로 말하는데 리용된다.

백색광원에 의하여 방출되는 빛에네르기는 가시주파수에서 그림 15-3과 같은 분포를 가진다. 붉은색으로부터 보라색까지의 범위에 있는 매개 주파수성분은 전체 에네르기에 대략 같게 기여하며 하여 광원의 빛색은 흰색으로 표현한다. 기본주파수가 있을 때 광원의 빛에네르기분포는 그림 15-4와 같은 형식을 취한다. 지금 빛은 기본주파수에 대응하는 색을 가진다고 표현할수 있다. 이 그림에서 빛의 주성분의 에네르기밀도는 E_D로 표시하며 다른 주파수성분들은 에네르기밀도가 E_w인 백색빛을 만든다. 빛의 밝기는 방출되는 총에네르기밀도를 주는 곡선아래의 면적으로 계산할수 있다. 순도는 E_D와 E_w의 차에 관계된다. 기본주파수의 에네르기밀도 E_D가 백색빛성분의 에네르기밀도 E_w에 비하여 크면 클수록 빛은 점점 더 순수하다. 빛은 E_w=0일 때 100%, E_w= E_D일 때 0%의 순도를 가진다.

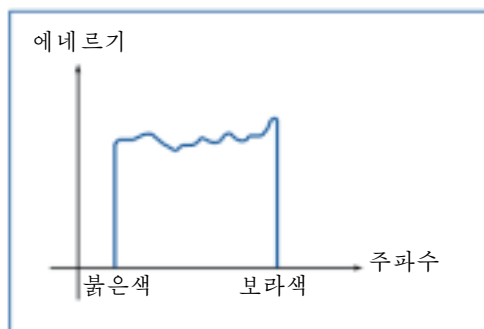



그림 15-3. 백색빛의 에네르기분포

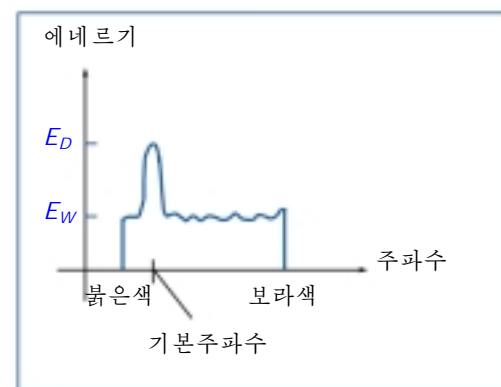



그림 15-4. 스펙트르의 붉은색부가까이에서 주주파수를 가지는 빛의 에네르기분포

둘이상의 광원들을 결합하면 본래의 광원들에 의하여 결정되는 특성량을 가지는 결과빛을 본다. 두 색광원의 세기를 적당히 조절하여 여러가지 다른 색들을 만들수 있다. 두 색광원을 결합하였을 때 백색빛이 만들어 지면 개개 색들을 보충색이라고 한다. 보충색쌍의 실례는 붉은색과 푸른풀색, 풀색과 분홍색, 푸른색과 누런색이다. 둘이상의 시작색들을 적중히 선택하여 넓은 범위의 다른 색들을 만들수 있다. 일반적으로 기본주파수(색채)에 의하여 빛의 결합을 표현하는 색모형들은 그 모형의 전색역(color gamut)이라고 하는 아주 넓은 범위의 색들을 얻기 위하여 세가지 색을 리용한다. 이러한 색모형에서 다른 색들을 만드는데 리용하는 둘 또는 세가지 색을 원색이라고 한다.

보이는 모든 색들을 만들기 위하여 결합할수 있는 실지의 원색들의 유한모임은 없다. 그러나 세가지 원색은 대부분의 목적들에 충분하며 지적된 원색모임의 전색역에 없는 색들은 여전히 확장된 방법에 의하여 표현할수 있다. 어떤 색을 세가지 원색의 결합으로 만들수 없다면 그 색을 만들기 위하여 세가지 원색의 결합에 하나 또는 두개의 다른 원색을 섞을수 있다. 이 확장된 의미에서 원색들의 모임은 모든 색을 표현한다고 고찰할수 있다. 그림 15-5에서는 임의의 스펙트르색을 만드는데 필요한 붉은색, 풀색, 푸른색량을 보여주었다. 색정합함수라고 부르는 그림 15-5의 곡선들은 대단히 많은 관측자들의 판단을 평균하여 얻었다. 500nm근방의 색들은 푸른색빛과 풀색빛의 결합에서 붉은색빛을 뽑아야만 만들수 있다. 이것은 500nm근방의 색은 그림에 지적된 푸른색풀색결합을 만들기 위하여 그 색을 붉은색빛과 결합시켜야만 표현된다는 것을 의미한다. 이리하여 RGB색현시장치는 500nm근방의 색을 현시할수 없다.

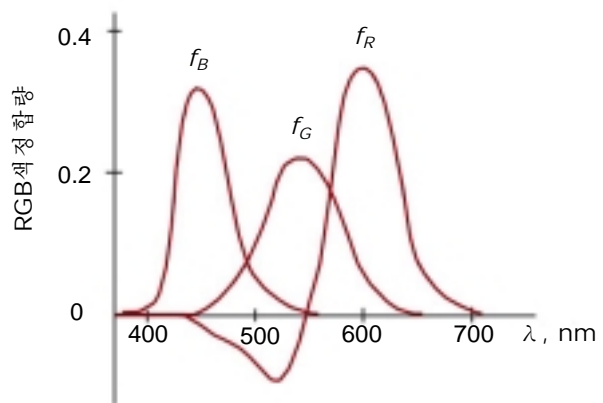



그림 15-5. 스펙트르색을 현시하는데 필요한 RGB 원색량

2절. 표준원색과 색도그림

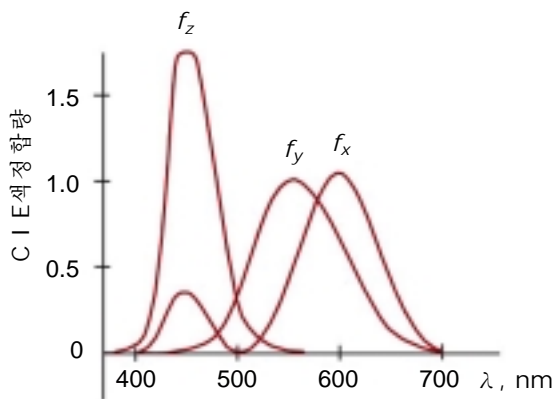



그림 15-6. 스펙트르색을 현시하는데 필요한 CIE 표준원색들의 량

가능한 모든 색들을 현시하기 위하여 결합할수 있는 색광원들의 유한모임이 없기때문에 국제조명위원회(CIE, Commission Internationale de 1'Éclairage)는 1931년에 세가지 표준원색을 정의하였다. 표준원색들은 가상적인 색들로서 그것들은 임의의 스펙트르색을 표현하는데 필요한 매개 원색의 량을 지적하는 정의 색정합함수들(그림 15-6)에 의하여 수학적으로 정의된다. CIE표준원색의 정의는 모든 색들에 대한 국제적인 표준정의를 제공하며 부값색정합과 실지 원색들의 선택에 관계되는 기타 문제들을 없앤다.

XYZ색모형

일반적으로 CIE표준원색들의 모임을 XYZ 또는 (X, Y, Z)색모형이라고 한다. 여기서 X, Y, Z는 3차원가색공간의 벡토르들을 표현한다. 임의의 색 C_λ는

$$C_\lambda = X\mathbf{X} + Y\mathbf{Y} + Z\mathbf{Z} \tag{15-2}$$

와 같이 표현된다. 여기서 X, Y, Z는 C_λ와 정합하는데 필요한 표준원색들의 량을 지적한다.

식 15-2의 표준원색들의 량을 휘도(X+Y+Z)에 대하여 정규화하면 색의 특성을 설명하는데 편리하다. 정규화된 표준원색량은

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = \frac{Z}{X+Y+Z} \tag{15-3}$$

와 같이 계산된다. 여기서 $x+y+z$=1이다. 이리하여 임의의 색은 x와 y량에 의하여 표현할수 있다. 휘도에 대하여 정규화한후 파라메터 x와 y는 색채와 순도에만 관계되기때문에 색도값이라고 한다. 색을 x와 y값에 의해서만 지적하면 량 X, Y, Z를 얻을수 없다. 따라서 색의 완전한 표현은 일반적으로 3개의 값 x, y, Y에 의하여 주어 진다. 나머지 CIE표준원색들의 량은

$$X = \frac{x}{y}Y, \qquad Z = \frac{z}{y}Y \tag{15-4}$$

와 같이 계산한다. 여기서 z=1－x－y이다. 색도자리표(x, y)를 리용하여 모든 색들을 2차원그림우에 표현할수 있다.

CIE색도그림

스펙트르색들의 정규화된 량 x, y를 점 찍어 표시하면 그림 15-7에 보여 준 혀모양의 곡선을 얻는다. 이 곡선으로 둘러막힌 그림을 CIE색도그림(chromaticity diagram)이라고 한다. 곡선의 점들은 전자기스펙트르에서의 순수한 색이며 스펙트르의 붉은색끝에서부터 보라색끝까지 nm파장에 따라 표식된다. 붉은색스펙트르점과 보라색스펙트르점을 련결하는 자주색선이라고 부르는 선은 스펙트르의 부분이 아니다.

내부의 점들은 보이는 모든 색들을 표현한다. 색도그림에서 점 C는 백색빛에 대응한다. 실제로 이 점은 C광원라고 알려 진 백색광원을 표시하며 이것은 평균일광에 대한 표준적인 근사로 리용된다.

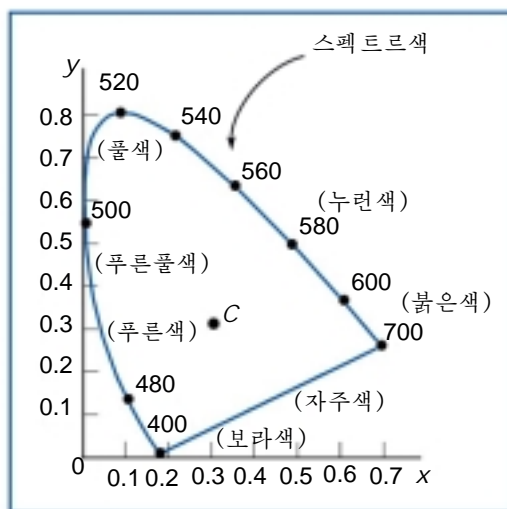



그림 15-7. CIE 색도그림에서 곡선우의 스펙트르색위치는 파장단위(nm)로 표식된다.

정규화때문에 휘도값은 색도그림에서 사용불가능하다. 휘도는 다르지만 동일한 색도를 가지는 색들은 동일한 점에 넘어 간다. 색도그림은 다음과 같은데 쓸모 있다.

- 서로 다른 원색모임들의 전색역을 비교하는데
- 보충색을 식별하는데
- 주어 진 색의 주파장과 순도를 결정하는데

전색역은 색도그림에서 선분이나 다각형으로 표현된다. 그림 15-8의 점 C_1와 C_2을 련결하는 선분의 모든 색들은 색 C_1와 C_2을 적당한 량 섞어 만들수 있다. C_1의 비률이 더 크면 결과색은 C_2보다 C_1에 더 가깝다. 그림 15-8의 C_3, C_4, C_5와 같은 3개의 점에 대한 전색역은 3개의 색위치들에서 정점을 가지는 3각형이다. 세가지 원색은 3각형의 내부나 변의 색들만을 만들며 색도그림안의 모든 색들을 포함할수 있는 3각형은 없다. 그러므로 보이는 모든 색들을 만들기 위하여 가법적으로 결합할수 있는 세가지 원색이 없다는것을 색도그림을 통하여 쉽게 알수 있다. 현시장치와 경복사장치들의 전색역은 색도그림우에서 편리하게 비교할수 있다.

두 점에 대한 전색역은 직선이기때문에 색도그림에서 보충색들은 C의 량쪽에 있는 직선련결되는 두 점으로 표현되여야 한다. 그림 15-9에서 두 색 C_1와 C_2을 적당한 량 섞을 때 백색빛을 얻을수 있다.

색의 주파장을 결정하기 위하여 두가지 원색에 대한 전색역의 보간을 리용할수 있다. 그림 15-10의 색점 C_1에 대하여 C로부터 C_1을 거쳐 점 C_s에서 스펙트르곡선과 사귀는 직선을 그을수 있다. 그러면 색 C_1는 백색 C와 스펙트르색 C_s의 결합으로 표현된다. 그러므로 C_1의 주파장은 C_s이다. 주파장을 결정하는 이 방법은 C와 자주색선사이에 있는 색점들에 대하여 맞지 않을것이다. 그림 15-10에서 C로부터 점 C_2을 거쳐 직선을 그으면 보이는 스펙트르에 없는 자주색선의 점 C_p에 이른다. 점 C_2은 비스펙트르색이라고 말하며 그의 주파장은 스펙트르곡선에 있는 C_p의 보충색(C_{sp})으로 취한다. 비스펙트르색은 자주색-분홍색범위에 있으며 감법적인 주파장을 가지는 스펙트르분포를 가진다. 비스펙트르색은 주스펙트르파장(C_{sp}와 같은)을 백색빛에서 뽑아 만든다.

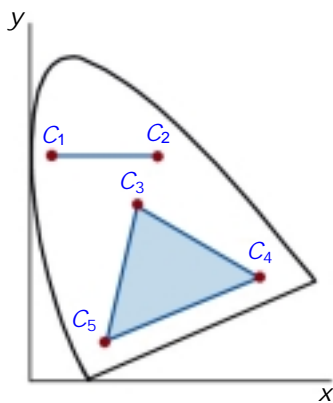



그림 15-8. 색도그림에서 원색들의 2색계와 3색계에 대하여 정의되는 전색역

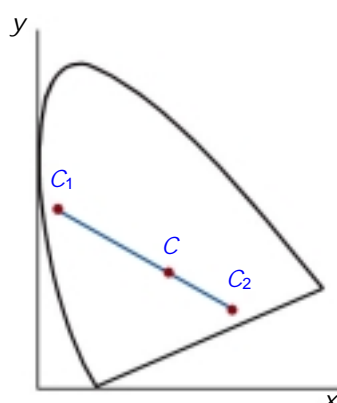



그림 15-9. 색도그림에서 보충색들의 표현

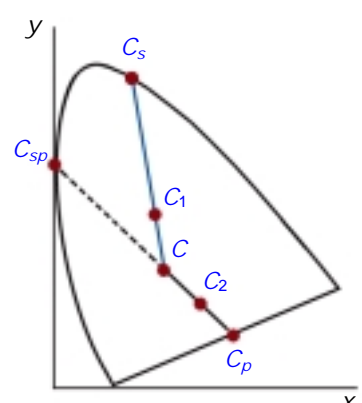



그림 15-10. 색도그림에 의한 주파장과 순도의 결정

그림 15-10의 C_1와 같은 임의의 색점에 대하여 순도는 C와 C_s를 련결하는 직선을 따라 C로부터의 C_1까지의 상대거리로 결정한다. d_{c1}가 C로부터 C_1까지의 거리를 표시하고 d_{cs}가 C로부터 C_s까지의 거리이면 순도는 비 d_{c1}/d_{cs}로 계산할수 있다. 이 그림에서 색 C_1는 C로부터 C_s까지의 전체 거리의 약 1/4에 위치하고 있기때문에 25%정도의 순도를 가진다. 위치 C_s에서 색점은 100%의 순도를 가진다.

3절. 색에 대한 직관적개념

미술가는 장면에 여러가지 명암(shade), 농담(tint), 색조(tone)를 만들기 위하여 색감에 검은 색감과 흰 색감을 섞어 색그림을 만든다. 순수한 색(또는 색채)의 여러가지 명암을 만들기 위하여 미술가는 그 색에 검은 색감을 추가한다. 검은 색감을 많이 추가할수록 명암은 점점 더 어두워진다. 류사하게 색의 여러가지 농담은 본래의 색에 흰 색감을 추가하여 얻을수 있으며 흰 색감을 많이 추가할수록 농담은 점점 더 연해 진다. 색의 여러가지 색조는 검은 색감과 흰 색감을 다같이 추가하여 만들수 있다.

이 색개념들은 색을 세가지 원색들의 상대적인 비률로 표현하는것보다 더 직관적이다. 일반적으로 색은 흰색을 추가하여 더 연하게 하고 검은색을 추가하여 더 어둡게 한다고 생각하는것이 훨씬 더 쉽다. 따라서 사용자에게 조색판을 제공하는 도형처리프로그람들은 자주 둘이상의 색모형을 쓴다. 한 모형은 사용자에게 직관적인 색선택대면부를 제공하고 다른것들은 출력장치에 대한 색성분들을 표현한다.

4절. RGB색모형

시각의 3자극리론에 기초하여 사람의 눈은 망막의 원추에서 3개의 시각적인 색자극을 통하여 색을 감각한다. 이 시각적인 색자극들은 약 630nm(붉은색), 530nm(풀색), 450nm(푸른색)의 파장에서 봉우리감도를 가진다. 색자극광원들의 세기를 비교하여 빛의 색을 감각한다. 현시장치에서는 이 시각리론에 기초하여 색을 현시하며 RGB색모형이라고 하는 세가지 원색 즉 붉은색, 풀색, 푸른색을 리용한다.

RGB색모형은 그림 15-11에 보여 준바와 같이 R, G, B 축에서 정의되는 단위바른6면체로 표현할수 있다. 원점은 검은색을 표현하며 자리표가 (1, 1, 1)인 정점은 흰색이다. 바른6면체의 R, G, B축정점들은 원색을 표현하며 나머지 정점들은 매개 원색의 보충색을 표현한다.

그림 15-11. 단위바른6면체에서 가색법에 의하여 색을 정의하는 RGB색모형

XYZ색모형과 마찬가지로 RGB색모형은 가법모형이며 여러가지 색들을 만들기 위하여 원색들을 더한다. 바른6면체안의 매개 색점은 (R, G, B)로 표현할수 있다. 여기서 R, G, B값은 0～1사이 범위에 있다. 그러므로 색 C_λ는 RGB 성분들로

$$C_\lambda = \mathrm{R}\mathbf{R} + \mathrm{G}\mathbf{G} + \mathrm{B}\mathbf{B} \qquad (15-5)$$

와 같이 표현된다. 분홍색정점은 쌍(1, 0, 1)을 만들도록 붉은색과 푸른색을 더하여 얻으며 흰색정점은 쌍(1, 1, 1)을 만들도록 붉은색, 풀색, 푸른색을 더하여 얻는다. 회색계조는 원점(검은색)으로부터 흰색정점까지 바른6면체의 주대각선을 따라 표현된다. 이 대각선의 매점에 대하여 원색들은 꼭같이 기여하며 그리하여 검은색과 흰색사이 중간의 무채색은 (0.5, 0.5, 0.5)로 표현된다. RGB바른6면체의 매면에서 색의 서서한 변화를 그림 15-12에 보여 주었다.

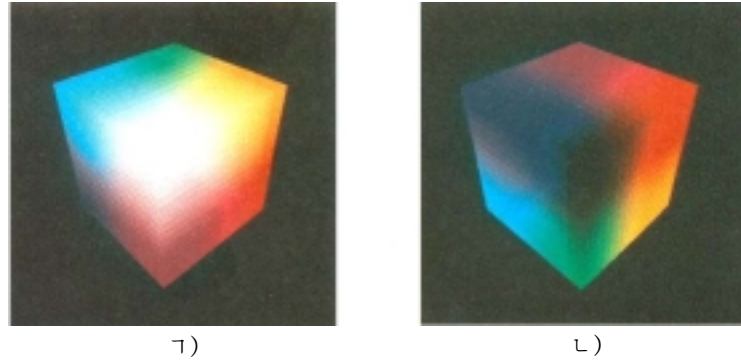



그림 15-12. RGB 바른6면체의 두가지 색보임상
ㄱ-흰색에서 검은색으로 회색계조대각선을 따라 본다.
ㄴ-검은색에서 흰색으로 회색계조대각선을 따라 본다.

NTSC표준RGB형광체의 색도자리표들을 표 15-1에 기입한다. 또한 CIE RGB색모형의 RGB색도자리표들과 색현시장치에서 형광체에 대하여 리용하는 근사값들을 기입한다. 그림 15-13에서는 NTSC표준RGB원색의 전색역을 보여 주었다.

표 15-1. RGB색도자리표(X, Y)

	*NTSC*표준	*CIE* 모형	색현시장치의 근사값
R	(0.670, 0.330)	(0.735, 0.265)	(0.628, 0.346)
G	(0.210, 0.710)	(0.274, 0.717)	(0.268, 0.588)
B	(0.140, 0.080)	(0.167, 0.009)	(0.150, 0.070)

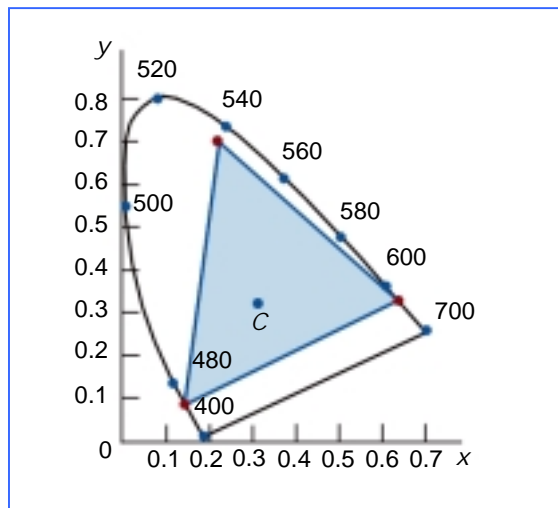



그림 15-13. RGB 전색역

5절. YIQ색모형

RGB현시장치는 화상의 붉은색, 풀색, 푸른색성분에 대한 개별신호를 요구하지만 텔레비죤은 단일합성신호를 리용한다. 합성영상신호를 만드는 NTSC의 색모형은 YIQ색모형이며 그것은 CIE XYZ색

모형에서의 개념들에 기초한다.

YIQ색모형에서 파라메터 Y는 XYZ색모형에서와 같다. 휘도(밝기)정보는 Y파라메터에 포함되며 한편 색도정보(색채와 순도)는 I파라메터와 Q파라메터에 병합된다. 표준시감도곡선을 만들기 위하여 Y파라메터에 대해 붉은색, 풀색, 푸른색세기들의 결합을 선택한다. Y는 휘도정보를 포함하기때문에 흑백텔레비죤은 Y신호만을 리용한다. NTSC영상신호에서 제일 큰 대역너비(약 4MHz)는 Y정보에 할당된다. 파라메터 Q는 약 0.6MHz대역너비에서 풀색-분홍색색채정보를 나른다.

RGB신호는 RGB값을 YIQ값으로 변환하고 다음에 I 및 Q정보를 Y신호에 변조하여 덧놓는 NTSC 부호기를 리용하여 텔레비죤신호로 변환할수 있다. RGB값으로부터 YIQ값에로의 변환은

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{15-6}$$

에 의하여 수행된다. 이 변환은 앞절에서 색도자리표가 주어 진 NTSC표준RGB형광체에 기초한다. 파라메터 Y에 할당되는 붉은색과 풀색의 비률이 푸른색에 비하여 더 크다는것은 밝기결정에서 이 색채들이 상대적으로 더 중요하다는것을 지적한다.

NTSC영상신호는 이 신호를 YIQ성분들로 가르고 다음에 RGB값으로 변환하는 NTSC복호기를 리용하여 RGB신호로 변환할수 있다. 식 15-6의 역행렬변환에 의하여 YIQ공간으로부터 RGB공간으로 변환한다.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.620 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.108 & 1.705 \end{bmatrix} \cdot \begin{bmatrix} Y \\ I \\ Q \end{bmatrix} \tag{15-7}$$

6절. CMY색모형

푸른풀색, 분홍색, 누런색을 원색으로 하여 정의하는 CMY색모형은 경복사장치에서 색출력을 표현하는데 쓸모 있다. 화면형광체들에서 나오는 빛을 결합하여 색무늬를 만드는 현시장치와 달리 작도기와 같은 경복사장치들은 종이에 색감을 칠하여 색그림을 만든다. 이때 감색반사빛의 색을 본다.

이미 언급한바와 같이 푸른풀색은 풀색빛과 푸른색빛을 더하여 만들수 있다. 따라서 흰색빛이 푸른풀색잉크로부터 반사될 때 반사빛에는 붉은색성분이 없다. 즉 붉은색빛은 잉크에 의하여 흡수되거나 빠진다. 류사하게 분홍색잉크는 입사빛으로부터 풀색성분을 뽑으며 누런색은 푸른색성분을 뽑는다. CMY색모형에 대한 단위바른6면체표현을 그림 15-14에서 보여 주었다.

그림 15-14. 단위바른6면체에서 감색법에 의하여 색을 정의하는 CMY 색모형

CMY색모형에서 점 (1, 1, 1)은 검은색을 표현한다. 왜냐하면 입사빛의 모든 성분들이 빠지기때문이다. 원점은 흰색을 표현한다. 같은 량의 원색들은 바른6면체의 주대각선을 따라 회색계조를 만든다. 푸른풀색잉크와 분홍색잉크를 결합하면 푸른색이 된다. 왜냐하면 입사빛의 붉은색성분과 풀색성분이 흡수되기때문이다. 류사하게 다른 색들도 감색법에 의하여 얻어 진다.

CMY색모형을 리용하는 인쇄공정은 RGB현시장치가 3개

의 형광체점들의 집합을 리용하는것과 약간 비슷하게 4개의 잉크점들의 집합으로 색점을 만든다. 매 원색(푸른풀색, 분홍색, 누런색)에 대하여 한 점씩 리용하며 나머지 한 점은 검은색점이다. 검은색점을 포함하는것은 일반적으로 푸른풀색, 분홍색, 누런색잉크들을 결합할 때 검은색대신에 어두운 무채색이 만들어 지기때문이다. 일부 작도기들은 세가지 원색잉크를 서로 덧분무하고 그것들이 마르기전에 섞이도록 하여 여러가지 색들을 만든다.

RGB표현으로부터 CMY표현에로의 변환은 행렬변환

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{15-8}$$

에 의하여 수행할수 있다. 여기서 단위렬벡토르는 RGB색모형에서의 흰색이다. 류사하게 행렬변환

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} C \\ M \\ Y \end{bmatrix} \tag{15-9}$$

에 의하여 CMY표현으로부터 RGB표현에로 변환한다. 여기서 단위렬벡토르는 CMY색모형에서의 검은색이다.

7절. HSV색모형

HSV색모형은 사용자에게 원색들의 집합보다도 더 직관적인 색표현을 제공한다. 사용자는 스펙트르색과 함께 여러가지 명암, 농담, 색조를 얻기 위하여 추가할 흰색과 검은색의 량을 선택함으로써 색을 지적한다. 이 모형에서 색파라메터들은 색채(H), 포화도(S), 명암도(V)이다.

HSV색모형의 3차원표현은 RGB바른6면체로부터 유도된다. 흰색정점에서 원점(검은색)으로 대각선을 따라 바른6면체를 본다고 하면 그림 15-15에 보여 준 6각형형태를 가지는 바른6면체의 륜곽선을 보게 된다. 6각형의 테두리는 여러가지 색채를 표현하며 그것은 HSV 6각추의 밑면으로 리용된다(그림 15-16). 6각추에서 포화도는 수평축을 따라 측정되며 명암도는 6각추의 중심에 있는 수직축을 따라 측정된다.

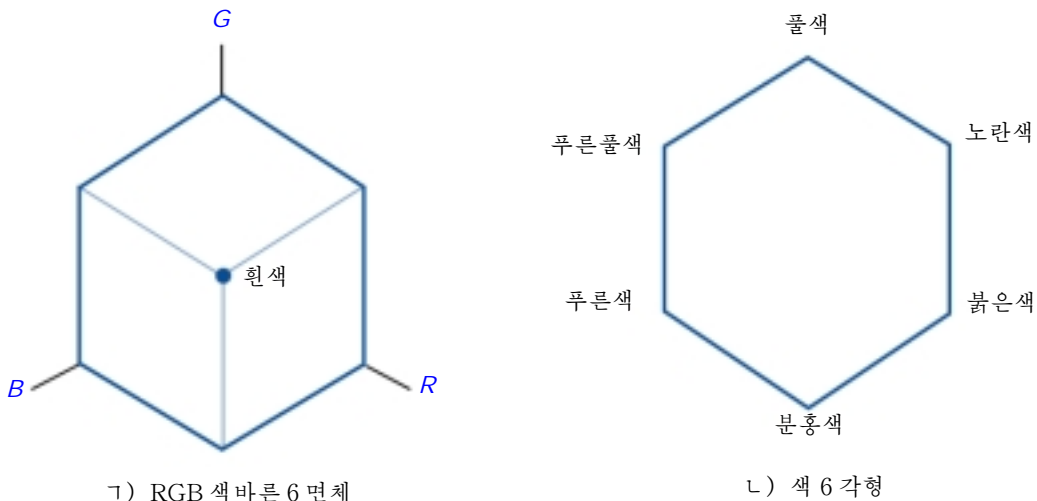



그림 15-15. RGB 바른 6 면체(ㄱ)를 흰색정점에서 원점으로 대각선을 따라 볼 때 바른 6 면체의 륜곽선은 6 각형(ㄴ)이다.

색채는 수직축에 대한 각으로 표현되며 0∼360° 범위에 있다. 6각형의 정점들은 60°간격으로 갈라 져 있다. 누런색은 60°, 풀색은 120°, 붉은색의 맞은편에 있는 푸른풀색은 180°위치에 있다. 보충색들은 180° 떨어 져 있다.

그림 15-16. HSV 6각추

포화도 S는 0∼1사이에서 변한다. 포화도는 이 모형에서 S=1에서의 최대순도에 대한 선택된 색의 순도의 비로 표현된다. 선택된 색은 S=0.25일 때 25%의 순도를 가진다고 말한다. S=0에서 회색계조를 가진다.

명암도 V는 6각추의 정점에서 0이고 밑면에서 1이며 그사이에서 변한다. 정점은 검은색을 표현한다. 6각추의 밑면에서 색채들은 자기의 최대세기를 가진다. V=1이고 S=1일 때 순수한 색을 가진다. 흰색은 V=1이고 S=0인 점이다.

HSV색모형은 대부분의 사용자들에게 보다 직관적인 색모형이다. 색채각 H를 지적하고 V=S=1로 설정하여 순수한 색을 선택한후 요구하는 색은 순수한 색에 검은색이나 흰색을 추가하여 얻는다. 검은색을 추가하기 위하여 V를 감소시키고 한편 S는 상수로 유지한다. 어두운 푸른색을 얻기 위하여 V는 0.4, S=1, H=240°로 설정할수 있다. 류사하게 흰색을 선택된 색채에 추가하기 위하여 파라메터 S는 감소시키고 한편 V는 상수로 보존한다. 연한 푸른색은 S=0.3, V=1, H=240°로 지적할수 있다. 검은색과 흰색을 함께 추가하기 위하여 V와 S를 다같이 감소시킨다. 이 모형의 대면부는 일반적으로 조색판에서 HSV파라메터들을 선택할수 있게 한다.

명암, 농담, 색조와 관련되는 색개념들은 HSV 6각추의 자름면에서 표현된다(그림 15-17). 순수한 색에 검은색을 추가하면 V는 6각추의 측면을 따라 감소된다. 그러므로 여러가지 명암은 값 S=1, $0 \le V \le 1$에 의하여 표현된다. 순수한 색에 흰색을 추가하면 6각추의 밑면을 따라 여러가지 농담을 만들수 있으며 이때 V=1, $0 \le S \le 1$이다. 여러가지 색조들은 검은색과 흰색을 다같이 추가하여 지적하며 이것은 6각추의 3각형자름면구역안의 색점들을 만든다.

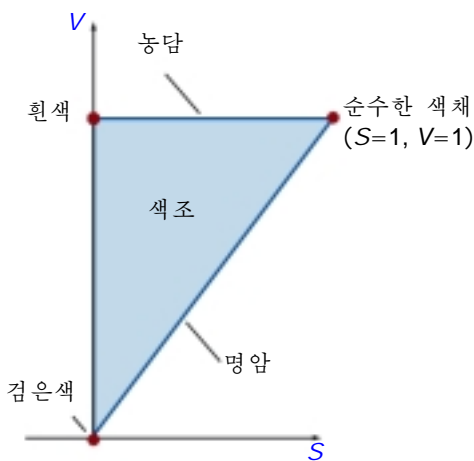
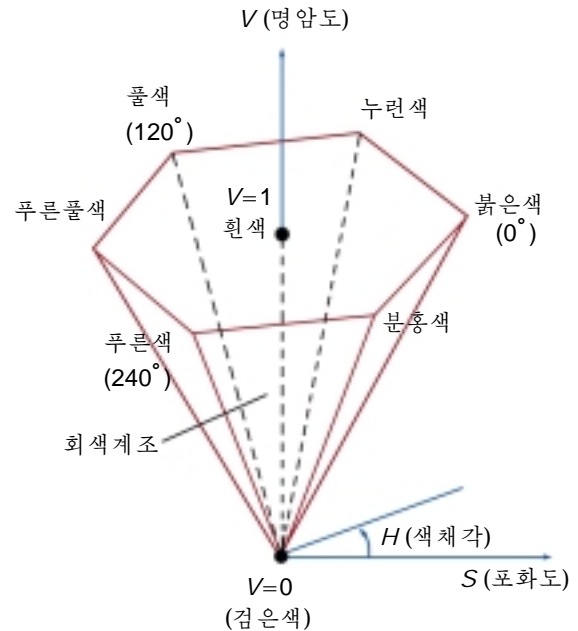



그림 15-17. 명암, 농담, 색조구역들을 보여 주는 HSV 6각추의 단면

사람의 눈은 128개정도의 색채와 130개정도의 농담(포화도준위)을 구별할수 있다. 이 매쌍에 대하여 색채의 선택에 따라 여러가지 명암들을 검출할수 있다. 누런색에서는 23개정도의 명암을 구별할수 있으며 스펙트르의 푸른색부에서는 16개정도의 명암을 가려 볼수 있다. 이것은 약 128×130×23=82720개의 색을 구별할수 있다는것을 의미한다. 대부분의 도형처리응용들에서는 128개의 색채, 8개의 포화도준위, 15개의 명암도준위이면 충분하다. HSV색모형에서 이 범위의 파라메터들에 의하여 16384개의 색이 사용자에게 사용가능하며 체계는 화소당 14bit의 색기억기를 요구한다. 화소당 기억기요구를 줄이고 사용가능한 색의 수를 증가시키기 위하여 색검색표를 리용할수 있다.

8절. HSV색모형과 RGB색모형사이의 변환

도형처리프로그람사용자가 HSV색파라메터들을 사용할 때 이 파라메터들은 색현시장치에 필요한 RGB값으로 변환된다. 이 변환에 필요한 조작을 결정하기 위하여 먼저 HSV각추를 RGB바른6면체로부터 어떻게 유도할수 있는가를 고찰한다. 이 바른6면체의 원점(검은색)으로부터 흰색정점까지의 대각선은 6각추의 *V*축에 대응한다. 또한 RGB바른6면체의 매개 부분바른6면체는 6각추의 6각형자름면구역에 대응한다. 임의의 6각형자름면에서 6각형의 모든 변들과 *V*축으로부터 임의의 정점까지의 모든 반경선들은 명암도 *V*를 가진다. 임의의 RGB값에 대하여 *V*는 그의 최대값과 같다. RGB값에 대응하는 HSV점은 명암도 *V*의 6각형자름면에 있다. 파라메터 *S*는 *V*축으로부터 이 점까지의 상대적인 거리로 결정된다. 파라메터 *H*는 6각형의 매개 6분구안에서 점의 상대적인 위치를 계산하여 결정한다. 임의의 RGB값을 대응하는 HSV값에로 넘기는 알고리듬을 다음의 절차에 준다.

```
#include <math.h>

#define MIN(a,b) (a<b?a:b)
#define MAX(a,b) (a>b?a:b)
#define NO_HUE  -1

/* Input:   r, g, b in range [0..1]
   Output: h, s, v in range [0..1]  */
void rgbToHsv (float r, float g, float b, float * h, float * s, float * v)
{
  float max = MAX (r, MAX (g, b)), min = MIN (r, MIN (g, b) );
  float delta = max - min;

  *v = max;
  if (max != 0.0)
    *s = delta / max;
  else
    *s = 0.0;
  if (*s == 0.0) *h = NO_HUE;
  else {
    if (r == max)
      *h = (g - b) / delta;
    else if (g == max)
      *h = 2 + (b - r) / delta;
    else if (b == max)
      *h = 4 + (r - g) / delta;
    *h *= 60.0;
    if (*h < 0) *h += 360.0;
    *h /= 360.0;
  }
}
```

HSV파라메터로부터 RGB파라메터를 얻는것은 우의 rgbToHsv절차에서 진행한 계산들을 반대로 진행하면 얻을수 있다. 이 역계산들은 6각추의 매개 6분구에 대하여 수행된다. 결과적인 변환식들을 다음의 알고리듬에서 개괄하였다.

```
#include <math. h>

/*  Input : h, s, v in range [0..1]
    Output: r, g, b in range [0..1] */

void hsvToRgb (float h, float s, float v, float * r, float * g, float * b)
{
  int i;
  float aa, bb, cc, f;

  if (s == 0) /* Grayscale */
    *r = *g = *b = v;
  else {
    if (h == 1.0) h = 0;
    h *= 6.0;
    i = ffloor (h);
    f = h - i;
    aa = v * (1 - s);
    bb = v * (1 - (s * f));
    cc = v * (1 - (s * (1 - f)));
    switch (i) {
    case 0: *r = v;  *g = cc; *b = aa; break;
    case 1: *r = bb; *g = v;  *b = aa; break;
    case 2: *r = aa; *g = v;  *b = cc; break;
    case 3: *r = aa; *g = bb; *b = v; break;
    case 4: *r = cc; *g = aa; *b = v; break;
    case 5: *r = v;  *g = aa; *b = bb; break;
    }
  }
}
```

9절. HLS색모형

직관적인 색파라메터에 기초하는 다른 색모형은 Tektronix가 리용하는 HLS색모형이다. 이 모형은 그림 15-18에 보여 준 2중원추표현을 가진다. 이 모형에서 3개의 색파라메터들은 색채(H), 밝기(L), 포화도(S)이다.

색채는 HSV색모형에서와 같은 의미를 가진다. 그것은 선택된 색채가 수직축에 대하여 위치하는 각이다. 이 모형에서 $H=0°$는 푸른색에 대응한다. 나머지 색들은 원추의 둘레에서 HSV모형에서와 같은 순서로 지적된다. 분홍색은 60°, 붉은색은 120°, 푸른풀색은 $H=180°$에 있다. 보충색들은 2중원추에서 180° 떨어 져 있다.

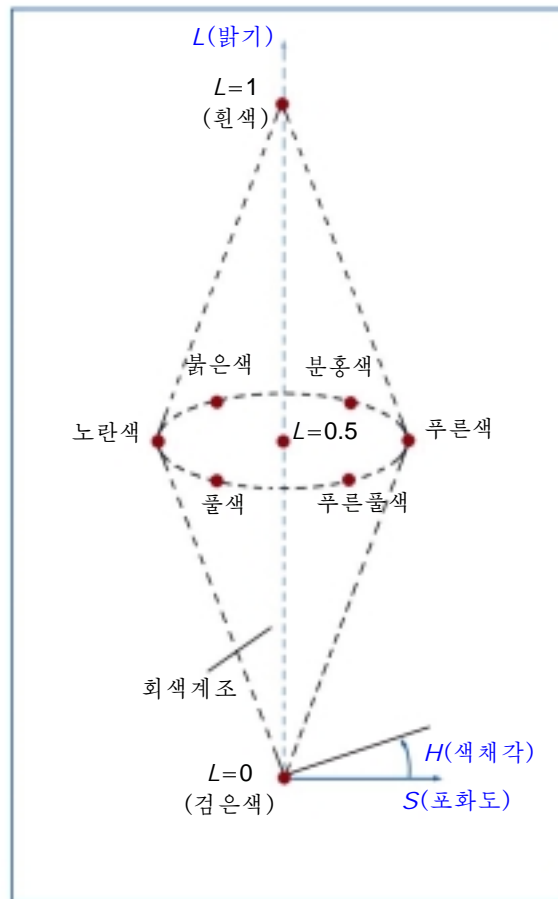



그림 15-18. HLS 2중원추

HLS색모형에서 수직축을 밝기 L이라고 한다. 검은색은 $L=0$에 있으며 흰색은 $L=1$에 있다. 회색계조는 L축을 따라 있으며 순수한 색채는 $L=0.5$인 면에 있다.

포화도파라메터 S는 색의 상대적인 순도를 지적한다. 이 파라메터는 0～1사이에서 변하며 순수한 색채는 $S=1$, $L=0.5$인것들이다. S가 감소할 때 색채는 덜 순수하다고 한다. $S=0$에서 회색계조를 가진다.

HSV색모형과 마찬가지로 HLS색모형은 사용자로 하여금 선택된 색채를 더 어둡게 하거나 또는 더 밝게 한다고 생각할수 있게 한다. 색채는 색채각 H에 의하여 선택하며 요구하는 명암, 농담, 색조는 L과 S을 조절하여 얻는다. 색은 L을 증가시켜 더 밝게 하고 L을 감소시켜 더 어둡게 한다. S가 감소될 때 색은 무채색에로 움직인다.

10절. 색의 선택과 응용

도형처리프로그람은 색선택을 쉽게 하여 색을 응용할수 있게 한다. 미끄럼띠와 색원을 리용하여 여러가지 색들을 선택할수 있으며 조화로운 색들을 쉽게 선택할수 있도록 체계를 설계할수 있다. 또한 프로그람설계자는 사용자에게 줄 색선택대면부를 설계할 때 몇가지 기초적인 색규칙들을 따를수 있다.

동위색들을 얻는 한가지 방법은 색모형의 부분공간에서 색들을 만드는것이다. 실례로 RGB 또는 CMY 바른6면체안의 임의의 직선을 따라 규칙적인 간격으로 색들을 선택하면 잘 정합되는 색들을 얻을수 있다. 우연적으로 색들을 선택하면 눈에 거슬리고 어울리지 않는 색들을 만들수 있다. 색선택에서는 또한 서로 다른 색이 서로 다른 깊이로 느껴 지는것을 고찰하여야 한다. 이것은 사람의 눈이 색의 주파수에 따라 주의를 집중하기때문에 일어 난다. 특히 푸른색은 희미해 지는 경향이 있다. 푸른색무늬를 붉은색무늬 다음에 현시하는것은 눈을 피로케 할수 있다. 왜냐하면 주의를 한 구역에서 다른 구역에로 돌릴 때 다시 초점맞추는것이 자주 일어 나기때문이다. 이 문제는 이 색들을 가르거나 또는 HSV색모형에서 색6각형의 절반 또는 그이하의 색들을 리용함으로써 줄일수 있다. 이 방법에 의하여 푸른색과 풀색 또는 붉은색과 누런색을 현시한다.

일반적으로 색들이 적을 때에는 많을 때보다 더 만족스럽게 현시되며 명암과 농담은 순수한 색채보다 더 잘 섞인다. 보통 배경에는 무채색이나 전경색의 보충색이 좋다.

요약

이 장에서는 빛의 기본특성과 색모형의 개념을 설명하였다. 보이는 빛은 전자기스펙트르안의 좁은 주파수대역이다. 빛은 기본주파수(또는 색채), 휘도(또는 밝기), 순도(또는 포화도)에 의하여 표현된다. 보충색은 백색빛을 만들도록 결합되는 색들이다.

색모형을 정의하는 한가지 방법은 여러가지 색들을 만들기 위하여 결합되는 둘이상의 원색들을 지적하는것이다. 세가지 원색에 의하여 정의되는 일반적인 색모형은 RGB색모형과 CMY색모형이다. 현시장치는 RGB색모형을 리용하며 한편 경복사장치들은 CMY색모형을 리용하여 색을 출력한다. 밝기값과 순도값의 지적에 기초하는 다른 색모형들에는 YIQ, HSV, HLS 색모형들이 있다. HSV 및 HLS 색모형과 같은 직관적인 색모형들은 색채값을 선택하고 선택된 색채에 추가할 흰색량과 검은색량을 선택하여 색을 지적할수 있게 한다.

유한원색모임에 의하여 지적되는 색모형으로는 가능한 모든 색들을 표현할수 없기때문에 CIE표준원색이라고 부르는 3개의 가상적인 색들을 모든 색표현의 표준으로 채용하였다. 일반적으로 CIE표준원색들을 XYZ색모형이라고 한다. 표준원색량 X와 Y의 정규화된 값들을 점 찍어 표시하면 임의의 색을 색채와 순도에 의하여 표현하는 CIE색도그림을 얻는다. 여러가지 색모형들의 전색역을 비교하고

보충색들을 식별하며 주어 진 색의 기본주파수와 순도를 결정하는데 색도그림을 리용할수 있다.

색현시에서 중요한것은 조화로운 색들을 선택하는것이다. 몇가지 간단한 규칙에 의하여 이것을 수행할수 있다. 동위색들은 보통 색모형의 작은 부분공간에서 선택할수 있다. 기본주파수가 크게 차이나는 색들을 련이어 현시하는것은 피해야 한다. 그리고 순수한 색채보다는 명암과 농담에 의하여 만들어 지는 적은 개수의 색들로 현시를 제한하여야 한다.

참고문헌

색에 대하여서는 Wyszecki와 Stiles(1982)에서 주고 있다. 색모형과 색현시방법들은 Durrett (1987), Hall(1989), Travis(1991)에서 설명된다. 여러가지 색응용알고리듬들은 Glassner(1990), Arvo (1991), Kirk(1992)에서 보시오. 인간의 시각체계와 빛과 색의 감각에 대한 추가적인 정보는 Glassner (1994)에서 보시오.

련습문제

15-1. RGB값을 HSV값으로 변환하는 식을 유도하시오.

15-2. HSV값을 RGB값으로 변환하는 식을 유도하시오.

15-3. 현시된 차림표로부터 HSV값을 선택하고 다음에 프레임완충기억기의 RGB값으로 변환하는 대화식절차를 쓰시오.

15-4. RGB값을 HLS값으로 변환하는 식을 유도하시오.

15-5. HLS값을 RGB값으로 변환하는 식을 유도하시오.

15-6. 색차림표로부터 HLS값을 대화식으로 선택하고 다음에 대응하는 RGB값으로 변환하는 프로그람을 작성하시오.

15-7. RGB공간에서 임의로 지적되는 두 위치사이에서 색들을 선형보간하여 만드는 프로그람을 작성하시오.

15-8. RGB공간의 지적된 부분공간에서 색들을 선택하는 대화식루틴을 쓰시오.

15-9. HSV공간에서 임의로 지적되는 두 위치사이에서 색들을 선형보간하여 만드는 프로그람을 작성하시오.

15-10. HLS공간에서 임의로 지적되는 두 위치사이에서 색들을 선형보간하여 만드는 프로그람을 작성하시오.

15-11. 현시장치에서 두개의 색격자를 나란히 현시하시오. 한 격자는 우연적으로 선택한 RGB 색들로 채우고 다른 격자는 작은 부분공간에서 선택한 색들로 채우시오. 여러가지 우연선택과 여러가지 RGB부분공간을 실험하고 두 색격자를 비교하시오.

15-12. HSV 또는 HLS색공간에서 색들을 선택하여 련습문제 15-11의 두개의 색격자를 현시하시오.

16장. 콤퓨터동화

일반적으로 콤퓨터로 만든 동화는 오락(움직이는 그림과 아동영화), 광고, 과학 및 공학연구, 훈련, 교육 등에서 리용되고 있다. 동화라고 하면 물체의 운동을 생각할수 있는데 **콤퓨터동화**(computer animation)란 일반적으로 장면에서 시간에 따르는 시각적인 변화들의 흐름을 말한다. 콤퓨터로 만든 동화에서는 평행이동과 회전에 의한 물체의 위치변화와 함께 물체의 크기, 색, 투명도, 겉문양의 시간에 따르는 변화도 현시할수 있다. 광고동화는 한 물체의 형태를 다른것으로 변화시킨다. 콤퓨터동화는 위치, 방향, 초점거리와 같은 카메라의 파라메터들을 변화시켜 만들수도 있고 광원을 움직이거나 조명 및 실감처리에 관계되는 파라메터들과 절차들을 변화시켜 만들수도 있다.

콤퓨터동화의 응용에서는 현실감이 있는 현시를 요구한다. 수값모형에 의하여 표현되는 우뢰비나 다른 자연현상들의 형태를 정밀하게 표현하는것은 그 모형의 믿음성을 평가하는데서 중요하다. 또한 비행기조종사와 중설비조작자를 위한 훈련모의기는 환경을 합리적으로 정밀하게 표현하여야 한다. 오락과 광고에서 시각효과는 더 흥미 있다. 과장된 형태나 비현실적인 운동과 변환장면들이 현시된다. 콤퓨터로 만든 정밀한 표현을 요구하는 오락 및 광고들도 많다. 그러나 일부 과학 및 공학연구에서는 현실감을 크게 중요시 하지 않는다. 실례로 연구사는 물리적과정의 본질을 쉽게 리해하기 위하여 물리적량을 시간에 따라 변하는 허위색 또는 추상적인 형태들로 현시한다.

1절. 동화의 설계

일반적으로 동화는 다음의 걸음들로 설계한다.

- 장면화판설계
- 물체정의
- 키프레임지적
- 중간프레임만들기

동화의 이 설계순서는 아동영화창작에서와 같다. 그러나 이 순서를 따르지 않는 응용들도 많다. 실례로 비행모의기의 실시간콤퓨터동화는 비행기를 조종하는데 따라 수값모형풀이에 의하여 가시화한 그림들을 현시한다. 프레임별동화(frame-by-frame animation)에서 장면의 매개 프레임은 개별적으로 생성 및 기억된다. 후에 프레임들은 필림에 기록될수 있으며 또는 실시간록화재생방식으로 련속 현시될수 있다.

장면화판은 동화의 줄거리이다. 장면화판은 동화를 일어 날 기초사건들의 모임으로 정의한다. 만들어 질 동화의 류형에 따라 장면화판은 사건의 거친 초벌그림들로 이루어 질수도 있고 사건의 기초사상들의 목록으로 작성될수도 있다.

물체정의는 동화에 관계되는 매 물체에 대하여 주어 진다. 물체는 다각형이나 스플라인과 같은 기초적인 형태들에 의하여 정의되며 그의 움직임도 함께 지적된다.

키프레임은 동화의 일정한 시간에서 장면의 세부화된 그림이다. 매개 키프레임에서 물체의 위치는 그 프레임의 시간에 따라 정해 진다. 일부 키프레임들은 동화의 시작과 끝에서 선택되며 다른것들은 키프레임들사이의 시간간격이 너무 커지지 않도록 간격 지어 진다. 복잡한 동화에서는 천천히 변하는 간단한 동화에서보다 더 많은 키프레임들이 지적된다.

중간프레임은 키프레임들사이의 프레임이다. 필요한 중간프레임의 개수는 동화현시에 리용될 매체에 의하여 결정된다. 필림은 초당 24프레임을 요구하며 도형처리말단장치는 초당 30~60프레임의 속도로 재생된다. 일반적으로 키프레임들사이의 시간간격은 키프레임의 매쌍에 대하여 3~5개의 중간프

레임이 있도록 설정된다. 동화의 현시속도에 따라 일부 키프레임들은 중복될수 있다. 중복이 없는 1분 필림흐름에 대하여 1440개의 프레임이 필요하다. 키프레임의 매쌍에 대하여 5개의 중간프레임이 있으면 288개의 키프레임이 필요하다. 동화가 그다지 복잡하지 않으면 키프레임들을 조금더 멀리 떨어 지게 간격 둘수 있다.

응용에 따라 운동의 검증과 편집, 록음테프의 생성과 동기화와 같은 여러가지 과제들이 제기될수 있다. 일반적인 동화를 만드는데 필요한 대부분의 함수들은 콤퓨터로 만든다. 그림 16-1과 16-2에서는 콤퓨터로 만든 동화의 프레임실례를 보여 주었다.

그림 16-1. 상을 탄 콤퓨터동화식짧은영화 *Luxo Jr* 의 한개 프레임(영화는 현실감 있는 탁상등을 만들기 위하여 키프레임동화체계와 아동영화의 동화기술을 리용하여 설계되였다. 마지막화상은 다중광원과 수속적인 결문양입히기기술에 의해 실감처리되였다.)

그림 16-2. 오스카상을 받은 첫 콤퓨터동화식짧은영화 Tin Toy 의 한개 프레임(영화는 키프레임동화체계를 리용하여 설계되였으며 넓은 얼굴표정모형화를 요구하였다. 마지막화상은 수속적인 명암처리, 자기그림자처리기술, 운동흐림, 결문양입히기를 리용하여 실감처리되였다.)

2절. 일반적인 콤퓨터동화함수

동화의 개발에서 물체조작, 실감처리, 카메라움직이기, 중간프레임만들기와 같은 걸음들은 콤퓨터풀이에 잘 적응된다. 동화프로그람들은(실례로 Wavefront) 동화를 설계하고 개별적인 물체들을 처리하는 특수한 함수들을 제공한다.

동화프로그람들에서 사용가능한 한가지 함수는 물체자료의 기억 및 관리를 위한 함수이다. 물체의 형태와 기타 파라메터들은 자료기지에서 기억 및 갱신된다. 물체와 관련된 다른 함수들은 운동만들기와 실감처리를 위한 함수들이다. 운동은 2차원 또는 3차원변환을 리용하여 지적된 조건에 따라 만들수 있다. 다음에 보이는 면의 식별과 실감처리알고리듬의 적용에 표준함수들을 리용할수 있다.

다른 일반적인 함수는 카메라의 움직임을 모의하기 위한 함수이다. 표준적인 카메라움직이기는 확대축소보기, 상하좌우움직이기, 기울이기이다. 마지막에 키프레임들이 지적되면 중간프레임들을 자동적으로 만들수 있다.

3절. 라스터동화

라스터체계에서 라스터조작들을 리용하여 제한된 응용의 실시간동화를 만들수 있다. 5장 8절에서 본바와 같이 *xy*평면에서 간단한 평행이동방법은 직4각형화소블로크를 한 위치에서 다른 위치로 이동

시키는것이다. 90°의 배수의 2차원회전 역시 간단히 수행되며 경계허상제거절차를 리용하면 직4각형 화소블로크를 임의의 각으로 회전시킬수 있다. 화소블로크를 회전시키자면 회전되는 블로크와 겹치는 화소들의 구역적용범위퍼센트를 결정하여야 한다. 동화를 투영면에서의 운동으로 제한하는 한 2차원 또는 3차원물체들의 실시간동화를 만들기 위하여 라스터조작들을 순차실행할수 있으며 이때에는 보기알고리듬이나 보이는 면 알고리듬을 실시할 필요가 없다.

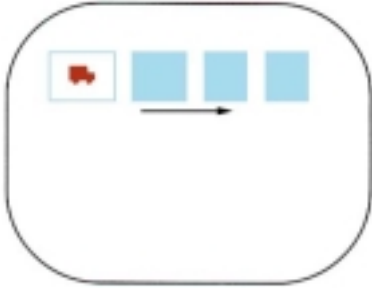



그림 16-3. 실시간라스터색표동화

또한 색표변환(color-table transformation)을 리용하여 물체를 2차원운동경로를 따라 움직이게 할수 있다. 운동경로의 매 위치들에서 미리 정의한 물체화소블로크들을 련속 색표서술항들에 대응시킨다. 첫번째 위치의 물체화소색서술항을 on값으로 설정하고 다른 위치의 물체화소색서술항들은 배경색으로 설정한다. 동화는 앞위치의 물체화소색서술항을 배경색으로 설정하면서 동화경로를 따라 매 위치의 물체화소색서술항들이 on값이 되도록 색표값들을 변화시켜 수행된다(그림 16-3).

4절. 콤퓨터동화언어

동화함수들을 리용하여 동화를 설계하고 조종한다. C, Lisp, Pascal, FORTRAN과 같은 일반목적언어가 동화함수의 프로그람작성에 리용되는데 여러가지 전문화된 동화언어들도 개발되였다. 동화함수들에는 도형편집함수, 키프레임발생함수, 중간프레임발생함수, 표준도형처리함수들이 있다. 도형편집함수는 스플라인곡면, 립체구성기하방법 기타 표현계획들을 리용하여 물체의 형태를 설계 및 수정할수 있게 한다.

동화지적에서 일반적인 과제는 장면표현이다. 여기서는 물체와 광원의 위치를 선정하고 광도측정파라메터(광원의 세기와 면조명특성량)들을 정의하며 카메라파라메터(위치, 방향, 렌즈특성량)들을 설정한다. 다른 표준함수는 동작지적이다. 여기에는 물체 및 카메라에 대한 운동경로의 설계가 속한다. 그리고 보통의 도형처리루틴들 즉 보기변환과 원근변환, 가속도함수나 운동학적경로지적함수와 같이 물체를 이동시키는 기하학적변환, 보이는 면식별, 면실감처리조작들이 필요하다.

키프레임체계(key-frame system)는 사용자가 지적하는 키프레임들로부터 중간프레임들을 만들기 위하여 간단히 설계된 전문화된 동화언어이다. 보통 장면안의 매개 물체는 제한된 개수의 자유도를 가지는 관절련결된 강체들의 모임으로 정의된다. 실례로 그림 16-4의 한팔로보트는 6개의 자유도를 가지는데 그것들을 팔회전, 팔뚝회전, 앞팔회전, 손목구부림, 손목선회, 손목회전이라고 한다. 이 로보트팔의 자유도의 개수는 받침대의 3차원평행이동에 의하여 9개까지 확장할수 있다(그림 16-5). 또한 받침대를 회전하게 하면 로보트팔은 총 12개의 자유도를 가질수 있다. 이에 비하여 인체는 약 200개의 자유도를 가진다.

파라메터체계(parameterized system)에서는 물체의 운동특성을 물체정의의 부분으로 지적할수 있다. 조정파라메터들은 자유도, 운동한계, 허용가능한 련속형태변화와 같은 물체특성들을 조종한다.

각본체계(scripting system)에서는 사용자입력각본에 의하여 물체와 동화를 정의할수 있다. 각본으로부터 여러가지 물체와 운동들의 서고를 만들수 있다.

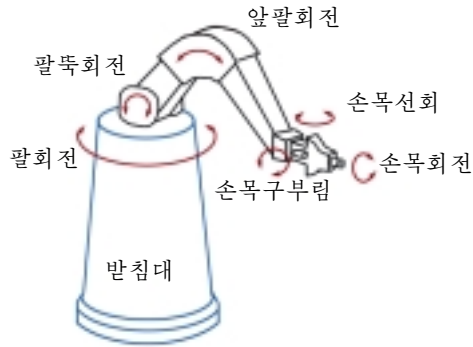



그림 16-4. 부동의 한팔로보트에 대한 자유도

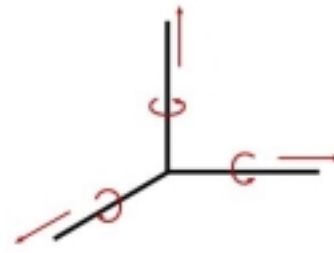



그림 16-5. 로보트팔의 받침대에 대한 평행이동자유도와 회전자유도

5절. 키프레임체계

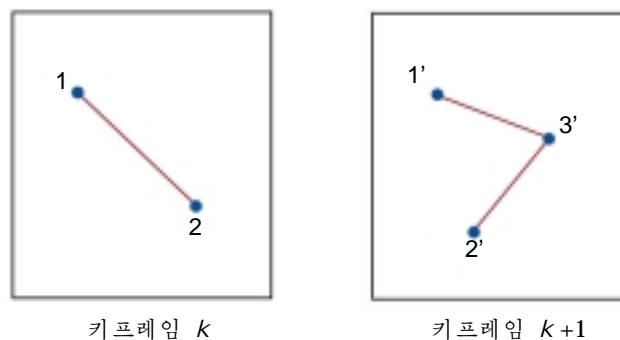
두개(또는 그이상)의 키프레임으로부터 중간프레임들을 만든다. 운동경로는 운동학적인 표현에 의해 스플라인곡선들의 모임으로 주어 질수 있다. 운동은 움직이는 물체에 작용하는 힘들의 지적에 의해 물리학에 기초할수 있다.

복잡한 장면에서 프레임들은 아동영화창작에서 쓰는 투명필림화(cels, celluloid transparencies)와 같은 개별적인 요소 또는 물체들로 가를수 있다. 동화경로가 주어 지면 임의의 두 시간사이에서 개별적인 물체들의 위치를 보간할수 있다.

복잡한 물체변환에 의하여 물체의 형태는 시간에 따라 변할수 있다. 실례로 천, 얼굴생김, 확대되는 세부, 형태의 발달, 물체의 폭발이나 분렬, 한 물체로부터 다른 물체에로의 변환 등을 들수 있다. 모든 면들이 다각형그물로 표현될 때 다각형당 변의 개수는 프레임마다 변할수 있으며 따라서 선분의 총 개수는 프레임마다 서로 다를수 있다.

련속형태변화

물체의 형태를 한 형식으로부터 다른 형식으로 변환하는것을 **련속형태변화**(morphing)라고 한다. 이것은 변태의 축소된 형식이다. 련속형태변화방법은 형태변화가 있는 임의의 운동이나 전이에 적용할수 있다.

물체변환에서 두개의 키프레임이 주어 지면 다각형의 변의 개수(또는 정점의 개수)가 두 프레임에서 같도록 한 프레임의 물체지적을 먼저 조정한다. 이 앞처리걸음을 그림 16-6에서 보여 주었다.

키프레임 k

키프레임 $k+1$

그림 16-6. 키프레임 k에서 정점위치 1과 2를 가지는 변은 키프레임 $k+1$에서 2련결변으로 변화한다.

키프레임 k의 선분은 키프레임 k+1의 두개의 선분으로 변환된다. 키프레임 k+1은 여분의 정점을 가지기때문에 두개의 키프레임에서 정점(과 변)의 개수의 균형을 맞추기 위하여 키프레임 k에서 정점 1과 2사이에 정점을 추가한다. 중간프레임들을 만들기 위하여 선형보간을 리용해서 그림 16-7의 직선경로를 따라 키프레임 k의 추가된 정점을 정점 3′로 변화시킨다. 그림 16-8에서는 4각형에로 선형적으로 확장되는 3각형의 실례를 보여 주었다. 그림 16-9와 16-10에서는 텔레비존광고에서의 련속형태변화실례들을 보여 주었다.

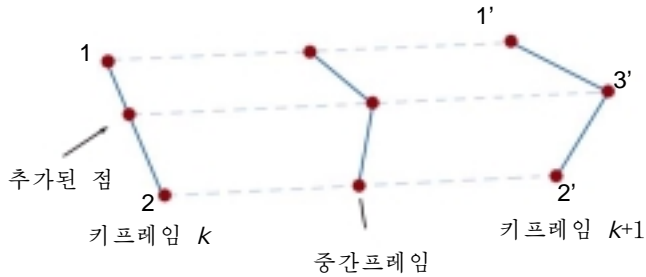



그림 16-7. 키프레임 k 의 선분을 키프레임 k+1 의 2 련결선분들로 변화시키는 선형보간

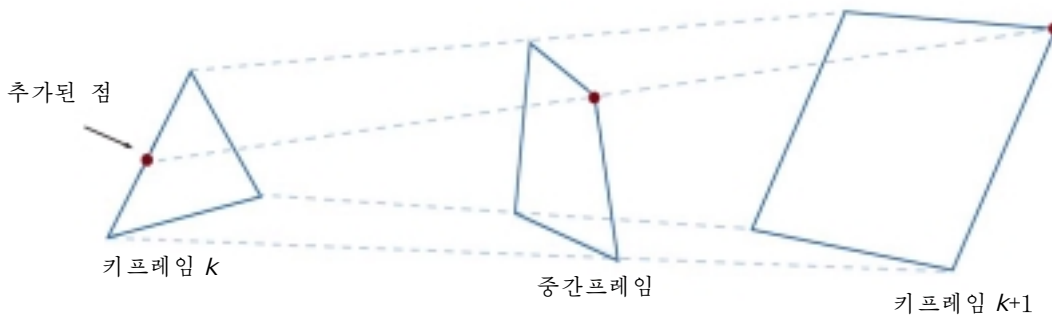



그림 16-8. 3 각형을 4 각형에로 변환하는 선형보간

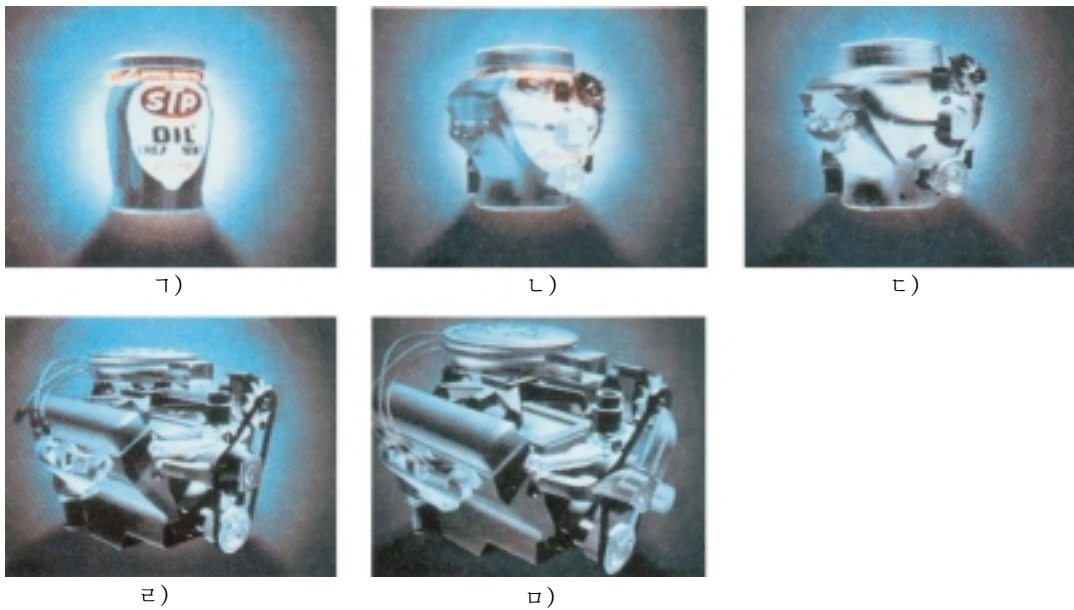



그림 16-9. STP 기름통을 기관블로크로 변환

ㄱ) ㄴ) ㄷ) ㄹ)

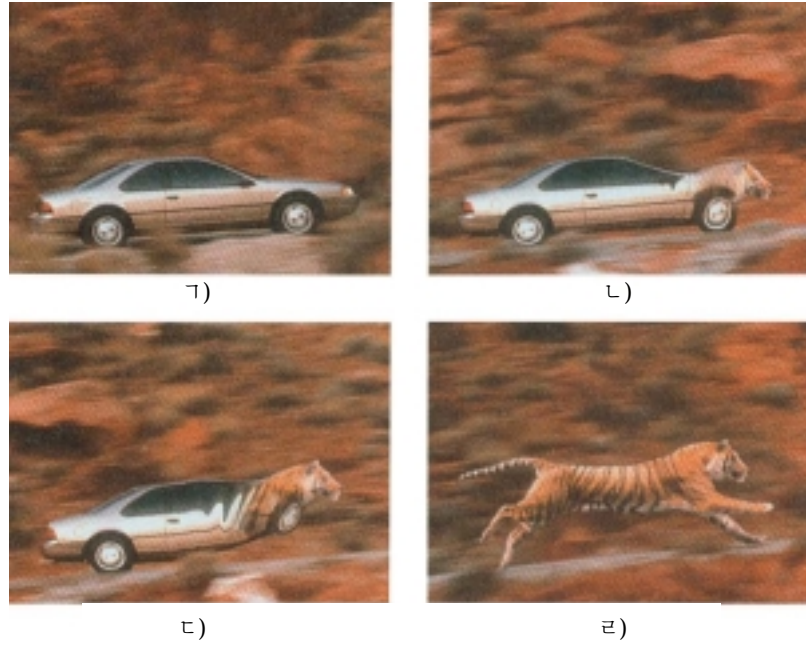



그림 16-10. 움직이는 자동차를 달리는 범으로 변환

키프레임에 변이나 정점을 추가하여 키프레임들을 같게 하는 일반적인 앞처리규칙을 말할수 있다. 변의 개수를 같게 하기 위하여 파라메터 L_k와 L_{k+1}은 두개의 이웃한 키프레임에서 선분의 개수를 표시한다고 하자. 그리고

$$L_{\max} = \max(L_k, L_{k+1}),\ \ L_{\min} = \min(L_k, L_{k+1}) \qquad (16-1)$$

$$N_e = L_{\max} \bmod L_{\min}$$
$$N_s = \operatorname{int}\left(\frac{L_{\max}}{L_{\min}}\right) \qquad (16-2)$$

로 정의한다. 그러면 앞처리는

1. 키프레임 min의 N_e개의 변들을 N_s+1개 부분들로 나눈다.
2. 키프레임 min의 나머지 선분들을 N_s개의 부분들로 나눈다.

에 의해 수행된다. 실례로 L_k=15, L_{k+1}=11이면 키프레임 k+1의 4개의 선분 매개를 2개의 부분들로 나누고 키프레임 k+1의 나머지 선분들은 그대로 남겨 둔다.

정점의 개수를 같게 하기 위하여 두개의 이웃한 키프레임의 정점의 개수를 파라메터 V_k와 V_{k+1}로 표시하자. 이 경우

$$V_{\max} = \max(V_k, V_{k+1}),\ \ V_{\min} = \min(V_k, V_{k+1}) \qquad (16-3)$$

그리고

$$N_{ls} = (V_{\max} - 1) \bmod (V_{\min} - 1)$$
$$N_p = \operatorname{int}\left(\frac{V_{\max} - 1}{V_{\min} - 1}\right) \qquad (16-4)$$

로 정의한다. 정점의 개수를 리용하는 앞처리는

1. 키프레임 min의 N_{ls}개 선분들에 N_p개의 점을 추가한다.
2. 키프레임 min의 나머지 변들에 N_p-1개의 점을 추가한다.

에 의해 수행된다. 3각형, 4각형실례에서 V_k=3, V_{k+1}=4이다. N_{ls}와 N_p는 다같이 1이며 따라서 키프레임 k의 한개 변에 한개의 점을 추가한다. 키프레임 k의 나머지 선분들에는 점을 추가하지 않는다.

가속도모의

곡선맞추기기술은 키프레임들사이의 동화경로를 지적하는데 자주 리용된다. 키프레임들에서 정점들의 위치가 주어 지면 이 위치들을 선형 또는 비선형경로로 맞출수 있다. 그림 16-11에서는 키프레임들사이에서 정점위치들의 비선형맞추기를 보여 주었다. 이것은 중간프레임들의 탄도를 결정한다. 가속도를 모의하기 위하여 중간프레임들사이의 시간간격을 조정할수 있다.

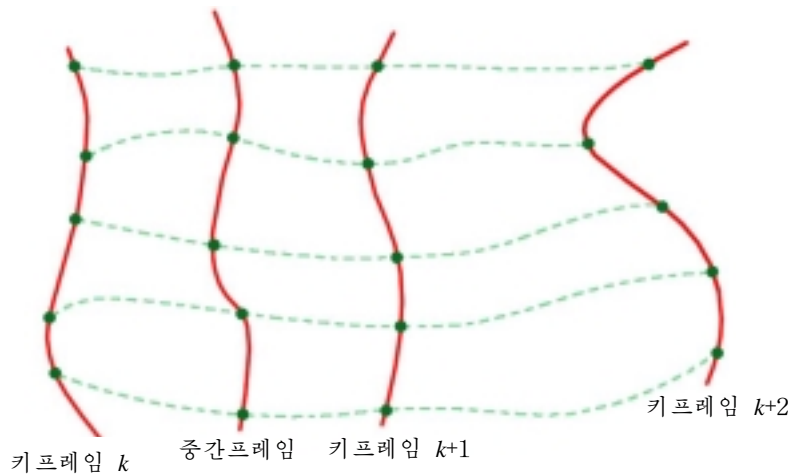



그림 16-11. 키프레임들사이에서 정점위치들을 비선형스플라인으로 맞추기

등속도(가속도가 령)일 때 중간프레임들에 대하여 등시간간격을 리용한다. 시간 t_1와 t_2의 키프레임쌍에 대하여 n개의 중간프레임들을 만든다고 하자(그림 16-12). 키프레임사이의 시간간격은 n+1개의 부분구간들로 나뉘여 지며 중간프레임간격은

$$\Delta t = \frac{t_2 - t_1}{n+1} \tag{16-5}$$

이다. 임의의 중간프레임의 시간은

$$tB_j = t_1 + j\Delta t, \quad j = 1, 2, \dots, n \tag{16-6}$$

와 같이 계산할수 있으며 자리표위치, 색, 기타 물리적파라메터들의 값을 결정할수 있다.

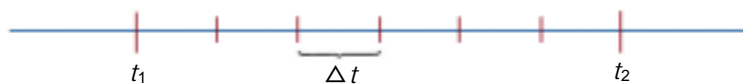



그림 16-12. 등속운동에 대한 중간프레임위치

비령가속도는 특히 동화의 시작과 끝에서 현실감이 있는 속도변화를 현시하는데 리용된다. 동화의 시동부분과 감속부분을 스플라인함수나 삼각함수에 의하여 모형화할수 있다. 가속도모형화에 포물선함수나 3차함수를 리용하기도 하지만 삼각함수는 동화프로그람들에서 보다 일반적으로 리용된다.

증가하는 속도(정의 가속도)를 모형화하기 위하여 물체가 더 빨리 움직일 때 더 큰 위치변화가 일어 나도록 프레임들사이의 시간간격을 증가시킨다. 증가하는 시간간격은 함수

$$1-\cos\theta, \qquad 0<\theta<\pi/2$$

에 의하여 얻을수 있다. n개의 중간프레임에 대하여 j번째 중간프레임의 시간은

$$tB_j = t_1 + \Delta t\left[1-\cos\frac{j\pi}{2(n+1)}\right], \qquad j=1,2,\ldots,n \qquad (16-7)$$

와 같이 계산된다. 여기서 Δt는 두개의 키프레임사이의 시간차이다. 그림 16-13에서는 삼각가속도함수의 곡선과 n=5에 대한 중간프레임간격을 보여 주었다.

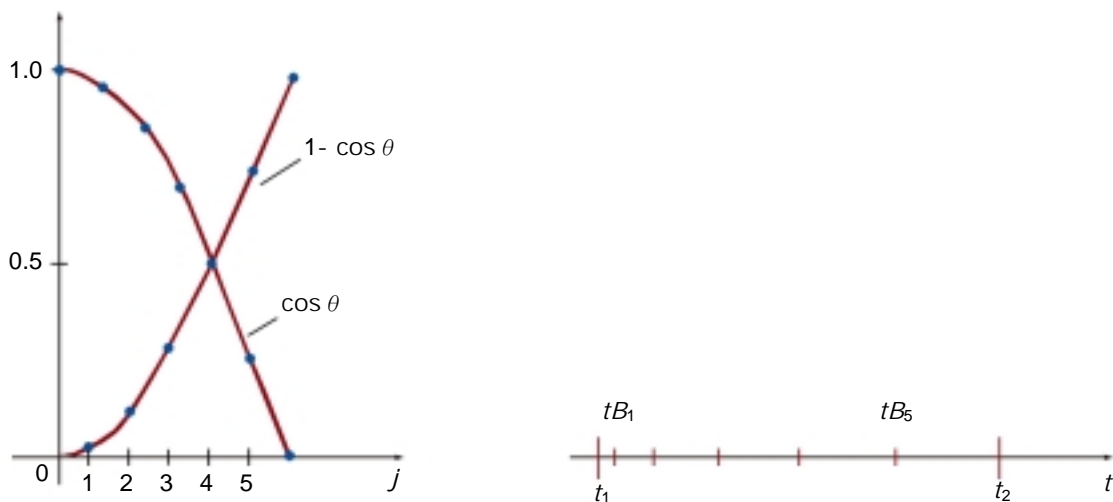



그림 16-13. 삼각가속도함수와 식 16-7에서 n=5 및 $\theta=j\pi/12$에 대한 중간프레임간격
(물체가 매 시간구간에서 움직일 때 자리표변화는 증가한다.)

감소하는 속도(감속도)는 $0<\theta<\pi/2$범위의 $\sin\theta$에 의해 모형화할수 있다. 중간프레임의 시간은

$$tB_j = t_1 + \Delta t\sin\frac{j\pi}{2(n+1)}, \qquad j=1,2,\ldots,n \qquad (16-8)$$

와 같이 정의된다. 그림 16-14에서는 이 함수의 곡선과 5개의 중간프레임에 대한 감소하는 시간간격을 보여 주었다.

운동에는 가속과 감속이 다같이 속한다. 증가하는 속도와 감소하는 속도의 결합은 먼저 중간프레임들의 시간간격을 증가시키고 다음에 이 간격을 감소시켜 모형화할수 있다. 이 시간변화를 수행하는 함수는

$$\frac{1}{2}(1-\cos\theta), \quad 0<\theta<\pi/2$$

이다. j번째 중간프레임의 시간은

$$tB_j = t_1 + \Delta t \left\{ \frac{1 - \cos[j\pi/(n+1)]}{2} \right\}, \qquad j = 1, 2, \dots, n \tag{16-9}$$

와 같이 계산된다. 여기서 Δt는 두개의 키프레임사이의 시간차이다. 그림 16-15에 보여 준바와 같이 움직이는 물체에 대한 시간간격은 처음에 증가하고 다음에 감소한다.

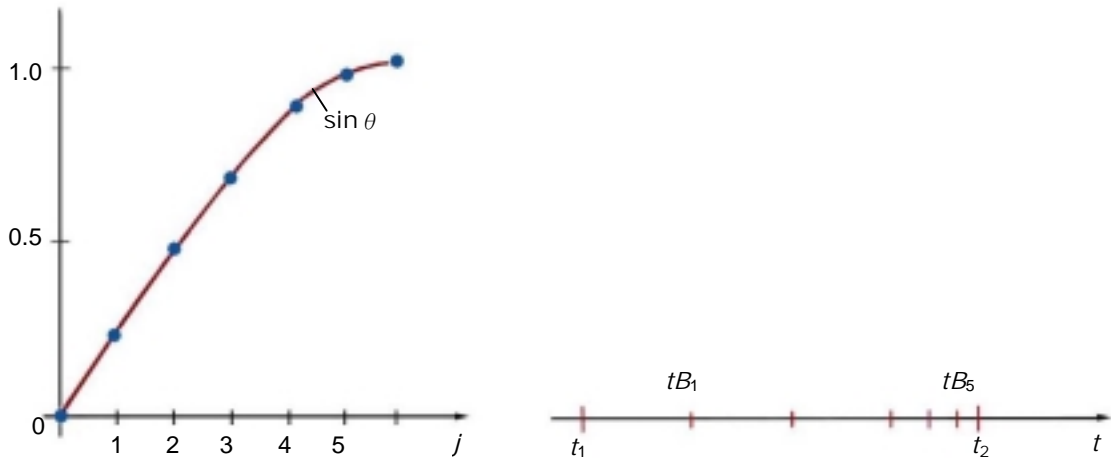



그림 16-14. 삼각감속도함수와 식 16-8 에서 n=5 및 $\theta = j\pi/12$ 에 대한 중간프레임간격
(물체가 매 시간구간에서 움직일 때 자리표변화는 감소한다.)

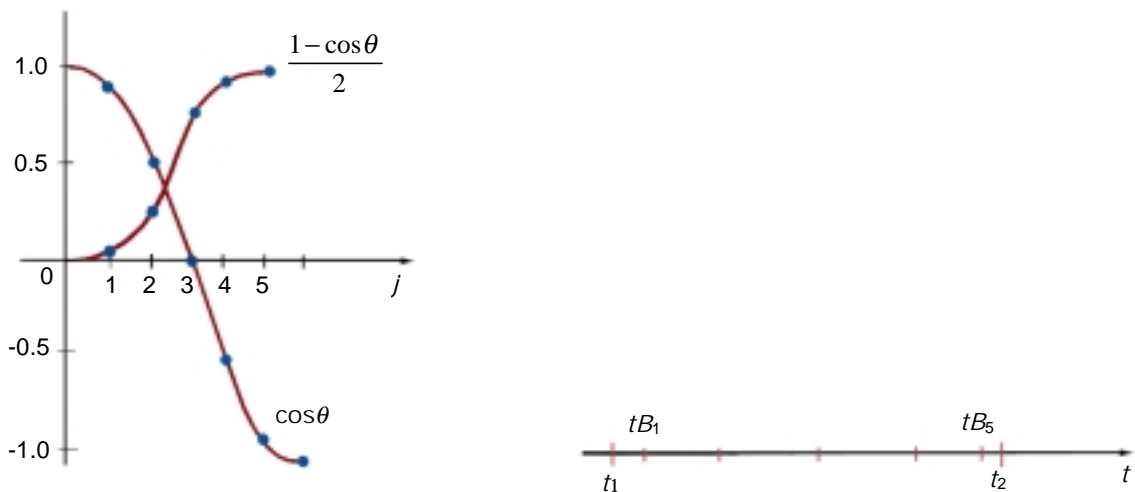



그림 16-15. 삼각가속감속함수와 식 16-9 에서 n=5 에 대한 중간프레임간격

초기에 물체들을 《골격》(선그물구조)모형화하여 중간프레임들을 간단히 처리한다. 이것은 동화를 대화식으로 조정할수 있게 한다. 동화가 다 정의된후 물체들을 완전히 실감처리할수 있다.

6절. 운동의 표현

동화체계에는 물체의 운동을 표현할수 있는 여러가지 방법들이 있다. 운동은 아주 명백히 정의할수도 있고 보다 추상적이거나 일반적인 방법들을 리용하여 정의할수도 있다.

운동의 직접표현

운동흐름을 정의하는 제일 간단한 방법은 운동파라메터들을 직접 지적하는것이다. 여기서는 회전각과 평행이동벡토르를 명백히 주고 자리표위치들을 변화시키는데 기하학적변환행렬들을 적용한다. 또한 일정한 종류의 운동을 지적하기 위하여 근사식을 리용할수도 있다. 실례로 튀여 오르는 공의 경로는 감쇠되는 수정된 시누스곡선

$$y(x) = A|\sin(\omega x + \theta_0)|e^{-kx} \qquad (16-10)$$

으로 근사할수 있다(그림 16-16). 여기서 A는 초기진폭, ω는 각주파수, θ_0은 위상각이다. 이 방법들은 사용자가 간단한 동화흐름의 프로그람을 작성할 때 리용될수 있다.

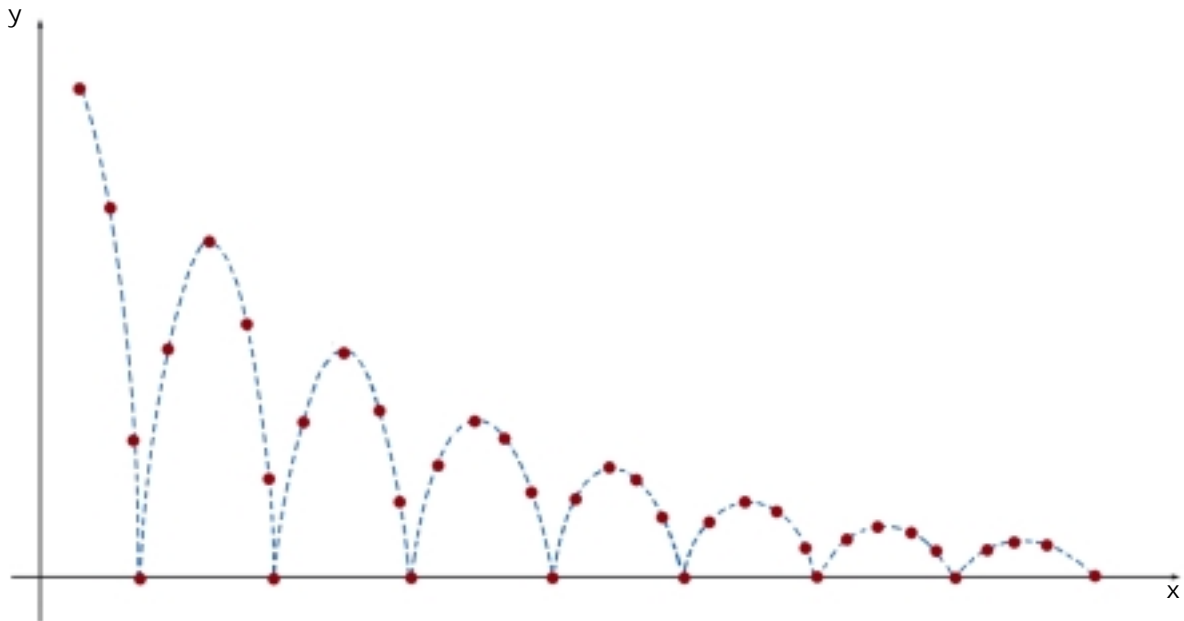



그림 16-16. 감쇠시누스함수(식 16-10)에 의한 튀여 오르는 공의 운동근사

목표지적법

운동의 직접표현과 반대로 동작을 추상적으로 표현하는 일반적인 용어들에 의하여 운동을 표현할수 있다. 이 방법은 동화의 목표가 주어 지면 구체적인 운동파라메터들을 결정하기때문에 목표지적법(goal-directed system)이라고 한다. 실례로 물체가 구체적인 목적지에로 걷기 또는 달리기할것을 바란다고 지적할수 있다. 또는 한 물체가 다른 물체를 좇기 바란다고 말할수 있다. 다음에 입력지령들은 선택된 과제를 수행하게 될 요소운동들로 해석된다. 실례로 사람의 운동은 몸통, 손과 발 등에 대한 요소운동들의 계층적인 구조로 정의할수 있다.

운동학과 동력학

동화는 운동학이나 동력학적인 표현을 리용하여서도 만들수 있다. 운동학적인 표현에서는 운동을 일으키는 힘에 대한 참조없이 운동파라메터(위치, 속도, 가속도)들을 주어 동화를 지적한다. 등속도(가속도가 령)일 때 장면안의 강체의 운동은 매 물체의 초기위치와 속도벡토르를 주어 지적한다. 실례로 속도가 (3, 0, -4)km/s로 지적되면 이 벡토르는 직선운동경로의 방향을 주며 속도의 크기는 5km/s이다. 또한 가속도(속도의 변화률)를 지적하면 가속, 감속곡선운동경로를 발생시킬수 있다. 운동학적인 운동지적은 간단히 운동경로를 표현하여서도 주어 질수 있다. 이때에는 자주 스플라인곡선을 리용한다.

또 다른 방법은 역운동학(inverse kinematics)을 리용하는것이다. 여기서는 물체의 초기 및 마지막위치를 지적하며 운동파라메터들은 체계에 의하여 계산된다. 실례로 령가속도를 가정하면 초기위치로부터 마지막위치까지 물체를 움직이게 될 등속도를 결정할수 있다.

다른 한편 동력학적인 표현에서는 속도와 가속도를 만드는 힘을 지적하여야 한다. 일반적으로 힘의 영향하에서 물체의 행동을 표현하는것을 물리학적모형화라고 한다(10장). 물체의 운동에 영향을 미치는 힘들의 실례는 전자기힘, 중력, 쓸림힘, 기타 력학적인 힘들이다.

물체의 운동은 중력과 마찰과정에 대한 뉴톤의 운동법칙, 류체흐름을 표현하는 오일레르 또는 나비에-스톡스의 방정식, 전자기힘에 대한 막스웰방정식과 같은 물리적인 법칙들을 표현하는 힘방정식들로부터 얻어 진다. 실례로 질량이 m인 립자에 대한 뉴톤의 제2법칙의 일반형식은

$$\mathbf{F} = \frac{d}{dt}(m\mathbf{v}) \tag{16-11}$$

이다. 여기서 **F**는 힘벡토르, **v**는 속도벡토르이다. 질량이 상수이면 방정식 $\mathbf{F}=m\mathbf{a}$를 푼다. 여기서 **a**는 가속도벡토르이다. 그렇지 않으면 질량은 상대성운동 또는 단위시간당 잴수 있는 량의 연료를 소비하는 우주운반선의 운동에서와 같이 시간의 함수이다. 또한 물체의 초기 및 마지막위치와 운동의 류형이 주어 지면 힘들을 얻기 위하여 역동력학(inverse dynamics)을 리용할수 있다.

물리학적모형화의 응용은 복잡한 강체계와 천 및 합성수지재료와 같은 비강체계를 포함한다. 일반적으로 동력학방정식들로부터 초기조건 또는 경계값들을 리용하여 운동파라메터들을 증분적으로 얻는데 수값방법들을 리용한다.

요약

콤퓨터동화는 장면화판, 물체정의, 키프레임들을 지적하여 설정할수 있다. 장면화판은 동화의 줄거리이며 키프레임은 동화의 선택된 시간에서 물체운동의 세부를 정의한다. 키프레임들이 설정되면 한 키프레임으로부터 다음키프레임까지 원활한 운동을 만들기 위하여 중간프레임들을 만들수 있다. 콤퓨터동화는 장면에서 움직이는 카메라의 운동경로뿐아니라 장면안의 물체에 대한 운동도 지적할수 있다. 콤퓨터동화체계에는 키프레임체계, 파라메터체계, 각본체계가 있다. 2차원동화에 대하여 5장에서 설명한 라스터동화방법을 리용할수 있다.

일부 응용들에서 키프레임들은 한 물체의 형태를 다른것으로 변화시키는 련속형태변화흐름의 걸음들을 정의하는데 리용된다. 운동의 가속도와 감속도를 모의하기 위하여 중간프레임들사이의 시간간격을 변화시킨다.

평행이동파라메터와 회전파라메터에 의하여 운동을 표현할수 있다. 운동은 식으로 표현할수도 있고 운동학파라메터나 동력학파라메터로 표현할수도 있다. 운동학적인 운동표현에서는 위치, 속도, 가속도를 지적한다. 동력학적인 운동표현에서는 장면안의 물체에 작용하는 힘들을 지적한다.

참고문헌

콤퓨터동화체계와 방법의 추가적인 정보는 Magnenat-Thalmann과 Thalmann(1985), Barzel(1992), Watt와 Watt(1992)에서 보시오. 동화응용알고리듬들은 Glassner(1990), Arvo(1991), Kirk(1992), Gascuel(1993), Ngo와 Marks(1993), van de Panne와 Fiume(1993), Snyder등(1993)에 주었다. 련속형태변화방법은 Beier와 Neely(1992), Hughes(1992), Kent, Carlson과 Parent(1992), Sederberg와 Greenwood(1992)에서 설명된다. PHIGS에서 동화방법의 설명은 Gaskins(1992)에 주었다.

련습문제

16-1. 한개 다면체의 동화에 대하여 장면화판과 키프레임들을 설계하시오.

16-2. 련습문제 16-1에서 지적된 키프레임들에 대하여 선형보간을 리용하여 중간프레임들을 만드는 프로그람을 작성하시오.

16-3. 둘이상의 움직이는 물체들을 포함하도록 련습문제 16-1의 동화를 확장하시오.

16-4. 련습문제 16-3의 키프레임들에 대하여 선형보간을 리용하여 중간프레임들을 만드는 프로그람을 작성하시오.

16-5. 구를 지적된 다면체로 변환하는 련속형태변화프로그람을 작성하시오.

16-6. 가속도를 포함하는 운동을 지적하고 식 16-7을 실현하시오.

16-7. 가속도와 감속도를 다같이 포함하는 동화를 지적하고 식 16-7과 16-8에 주어 진 중간프레임간격계산을 수행하시오.

16-8. 식 16-9의 가속감속계산을 수행하는 동화를 지적하시오.

16-9. 주어 진 직4각형구역안에서 채운 원의 선형적인 2차원운동을 모의하는 프로그람을 작성하시오. 원의 초기속도는 주어 지며 원은 입사각과 같은 반사각으로 벽에서 튀여 난다.

16-10. 련습문제 16-9에서 원의 경로를 자르기 위하여 직4각형의 한 변을 앞뒤로 움직일수 있는 짧은 선분으로 교체함으로써 프로그람을 공-채유희로 변환하시오. 유희는 원이 직4각형의 내부로부터 달아 날 때 끝난다. 초기입력파라메터는 원의 위치, 방향, 속도이다. 유희점수에는 채로 원을 막는 회수를 포함시킬수 있다.

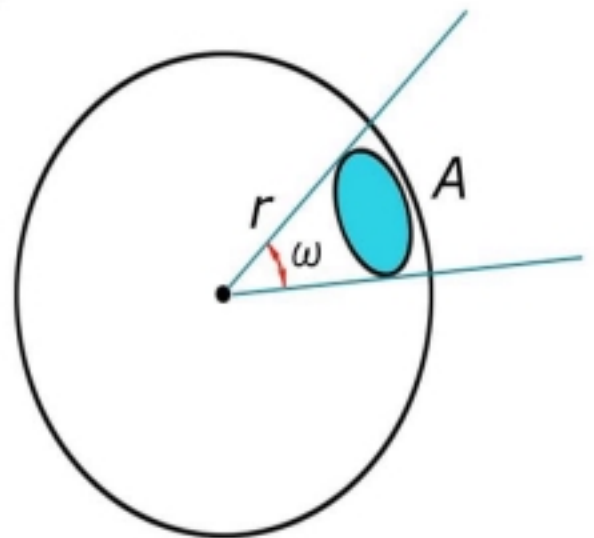
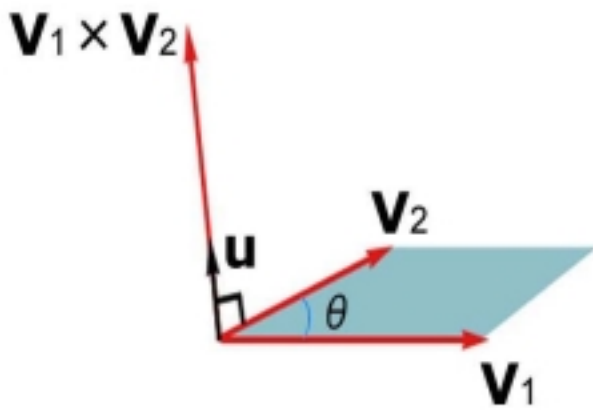
16-11. 련습문제 16-9의 프로그람을 평행6면체안에서 움직이는 구의 3차원운동을 모의하도록 확장하시오. 각이한 방향에서 운동을 보기 위하여 대화식보기파라메터들을 설정할수 있다.

16-12. 식 16-10을 리용하여 튀여 오르는 공을 모의하는 프로그람을 작성하시오.

16-13. 중력과 바닥면의 쓸림힘을 리용하여 튀여 오르는 공의 운동을 수행하는 프로그람을 작성하시오. 초기에 공은 주어 진 속도벡토르로 공간에 발사된다.

16-14. 2인용pillbox유희를 수행하는 프로그람을 작성하시오. 유희는 pillbox의 위치가 고정되였거나 우연지형특징을 가지는 평면에서 수행되며 pillbox는 유희를 시작할 때 배치된다.

16-15. 동력학적으로 운동을 지적하는 프로그람을 작성하시오. 둘이상의 물체, 초기운동파라메터, 주어 진 힘들로 장면을 지적하시오. 다음에 힘방정식들로부터 동화를 만드시오(실례로 물체는 질량에 정비례하고 거리의 두제곱에 역비례하는 인력을 가지는 지구, 달, 태양일수 있다.).

콤퓨터도형처리를 위한 수학

콤퓨터도형처리알고리듬들에서는 수학적인 개념들과 방법들을 많이 리용한다. 여기서는 이 책에서 설명한 도형처리알고리듬들에서 리용한 해석기하, 선형대수, 벡토르해석, 텐소르해석, 복소수, 수값해석 등 여러 분야들에 대한 간단한 지식을 준다.

부록 1. 자리표계

일반적으로 도형처리프로그람들에서 자리표파라메터들은 직각자리표계로 지적하는데 비직각자리표계도 쓸수 있다. 실례로 구대칭이나 원기둥대칭과 같은 대칭물체를 표현하거나 처리하는 식들을 간단화하는데는 비직각자리표계를 리용할수 있다. 그러나 전문적인 도형처리체계를 사용하지 않는 한 비직각자리표를 먼저 직각자리표로 변환하여야 한다. 이 절에서는 먼저 표준적인 직각자리표계를 복습하고 다음에 몇가지 일반적인 비직각자리표계들을 고찰한다.

2차원직각자리표계

그림 부-1에서는 두가지 가능한 직각화면자리표계를 보여 주었다. 자리표원점이 화면의 왼쪽 아래구석에 있는 그림 부-1 ㄱ의 자리표계는 일반적으로 리용되는 자리표계이다. 일부 체계들 특히 개인용콤퓨터들은 그림 부-1 ㄴ에서와 같이 자리표계의 원점을 화면의 왼쪽 웃구석에 놓는다. 일부 도형처리프로그람들은 자리표원점을 화면의 중심위치에 놓을수 있다.

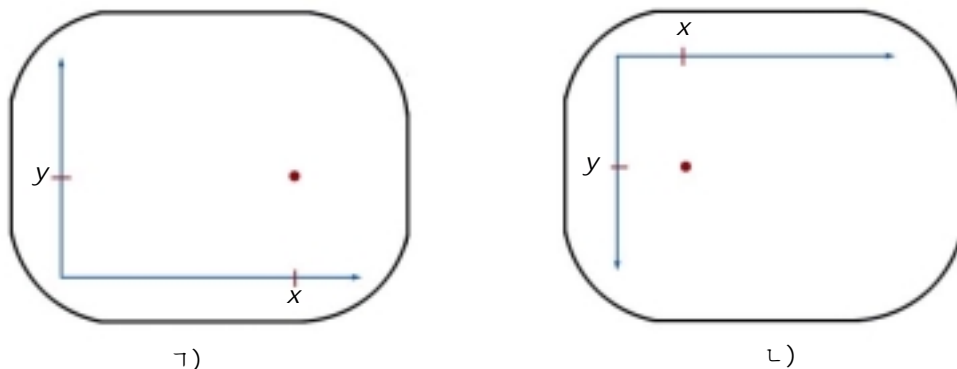



그림 부-1. 직각화면자리표계
ㄱ- 자리표원점이 화면의 왼쪽 아래구석에 있다.
ㄴ- 자리표원점이 화면의 왼쪽 웃구석에 있다.

*xy*평면에서의 극자리표계

극자리표계는 흔히 리용되는 비직각자리표계이다(그림 부-2). 자리표위치는 자리표원점으로부터의 반경거리 r와 수평선으로부터의 각변위 θ에 의하여 지적한다. 정의 각변위는 시계바늘반대방향이고 부의 각변위는 시계바늘방향으로 취한다. 각 θ는 도로 측정할수 있으며 원점주위로 시계바늘반대방향의 옹근 한 회전은 360°이다. 그림 부-3에서는 직각자리표와 극자리표사이의 관계를 보여 주었다. 그림 부-4의 직3각형에서 3각함수의 정의를 리용하면 식

$$x = r\cos\theta, \qquad y = r\sin\theta \qquad (부-1)$$

에 의해 극자리표가 직각자리표로 변환된다. 직각자리표로부터 극자리표에로의 역변환은

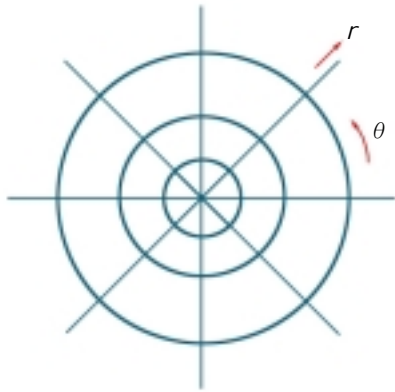



그림 부-2. 동심원과 반경선에 의해 형성되는 극자리표계

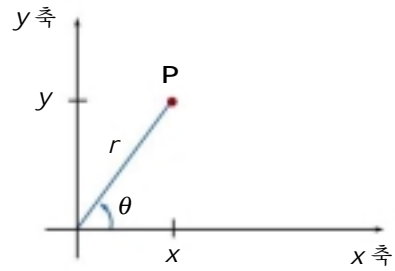



그림 부-3. 극자리표와 직각자리표

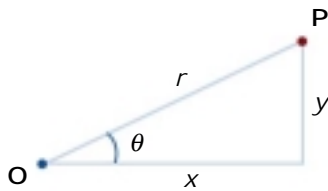



그림 부-4. 빗변 r, 직각변 x, y 를 가지는 직3각형

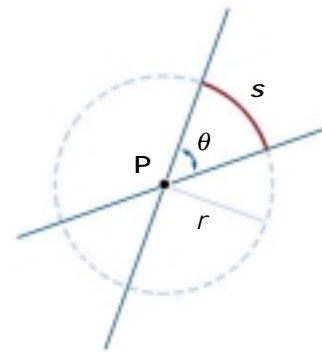



그림 부-5. 길이가 s 이고 반경이 r 인 원호가 마주하는 각 θ

$$r = \sqrt{x^2 + y^2}, \quad \theta = \tan^{-1}\left(\frac{y}{x}\right) \qquad (부-2)$$

자리표위치를 지적하는데는 원이외의 원추곡선들도 리용할수 있다. 실례로 원대신에 동심타원들을 리용하여 자리표위치를 타원자리표로 줄수 있다. 또한 쌍곡선이나 포물선의 자리표도 리용할수 있다.

각의 값은 도로도 지적할수 있고 무본단위(라디안)로도 지적할수 있다. 그림 부-5에서는 평면에서 사귀는 두선의 사귐점 **P**에 중심이 있는 원을 보여 주었다. 각 θ의 값은 라디안으로

$$\theta = \frac{s}{r} \qquad (부-3)$$

에 의하여 주어 진다. 여기서 s는 각 θ에 대한 원호의 길이이며 r는 원의 반경이다. 점 **P**주위의 전체각은 원둘레의 길이($2\pi r$)를 r로 나눈 2π라디안이다.

3차원직각자리표계

그림 부-6 ㄱ에는 3차원직각자리표계에서 자리표축들의 편리한 한가지 배치를 보여 주었다. 그림 부-6 ㄴ에 보여 준바와 같이 정의 x축으로부터 정의 y축방향으로 오른손 손가락들을 90° 구부려 z축을 움켜 쥘 때 오른손 엄지손가락이 정의 z축방향을 가리키기때문에 이 자리표계를 오른손자리표계라고 한다. 대부분의 콤퓨터도형처리프로그람들은 오른손직각자리표계에서 물체를 표현하고 처리할것을 요구한다. 이 책에서는(부록을 포함하여) 모든 직각자리표계를 오른손자리표계라고 가정한다.

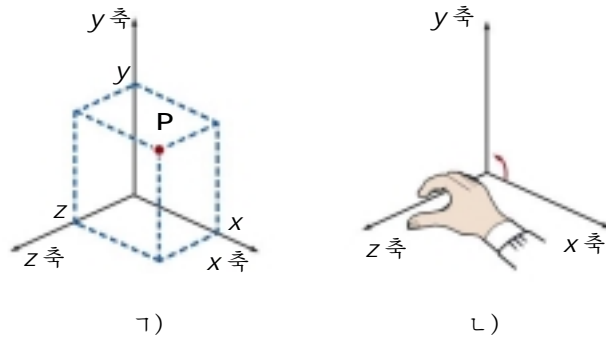



그림 부-6. 오른손직각자리표계에서 자리표가 (x, y, z)인 점 **P**의 위치표시

자리표축들을 달리 배치한 그림 부-7의 자리표계를 왼손자리표계라고 한다. 이 자리표계에서는 정의 x축으로부터 정의 y축방향으로 왼손 손가락들을 90° 구부려 z축을 움켜 쥘 때 왼손 엄지손가락이 정의 z축방향을 가리킨다. 자리표축들을 이렇게 배치하면 현시화면에서 물체들의 깊이를 표현하는데 편리하다. 자리표원점이 화면의 왼쪽 아래구석에 있는 왼손자리표계의 xy평면에서 화면위치를 표현하면 정의 z값은 그림 부-7 ㄱ에서와 같이 화면뒤의 위치를 지적한다. 정의 z축을 따라 z값이 크면 클수록 관찰자로부터 더 멀리에 있는것으로 된다.

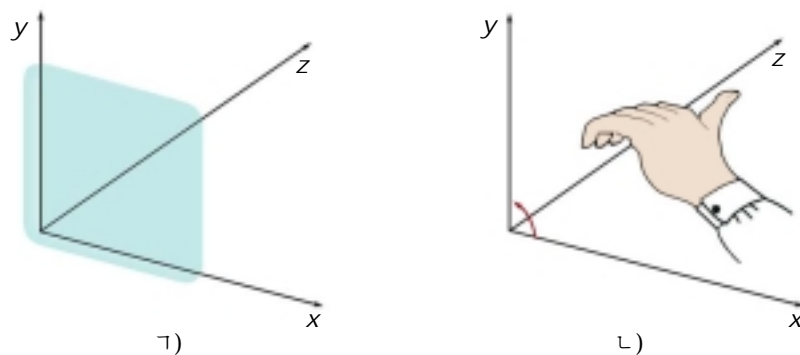



그림 부-7. 현시장치화면에 덧놓이는 왼손직각자리표계

3차원곡선자리표계

비직각자리표계를 곡선자리표계라고 한다.도형처리응용에서는 대칭, 계산의 쉬움, 가시화에서의 편리성 등을 고려하여 자리표계를 선택한다. 그림 부-8에서는 세개의 자리표면으로 형성되는 일반적인 곡선자리표계를 보여 주었다. 매면은 한개의 상수자리표를 가진다. 실례로 x_1x_2면은 상수자리표 x_3에 의하여 정의된다. 임의의 자리표계에서 자리표축들은 자리표면들이 사귀는 곡선이다. 자리표면들이 직각으로 사귈 때 직각곡선자리표계라고 한다. 빗각자리표계는 일반상대성법칙에 의하여 결정되는 운동의 가시화와 같은 특별한 목적에 쓸모 있다. 그러나 일반적으로 도형처리에서는 직각자리표계를 많이 리용한다.

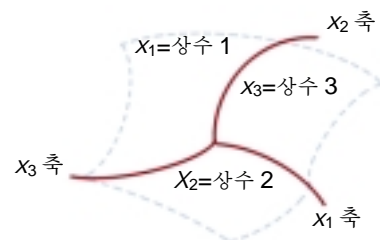



그림 부-8. 일반적인 곡선자리표계

그림 부-9에서는 원기둥자리표계에서 공간적인 위치의 지적을 직각자리표계와의 관계속에서 보여 주었다. 상수 ρ면은 수직원기둥, 상수 θ면은 z축을 포함하는 수직평면, 상수 z면은 xy평면에 평행인 수평평면이다. 계산

$$x=\rho\ \cos\theta,\quad y=\rho\ \sin\theta,\quad z=z \qquad (부-4)$$

에 의해 원기둥자리표를 직각자리표로 변환한다.

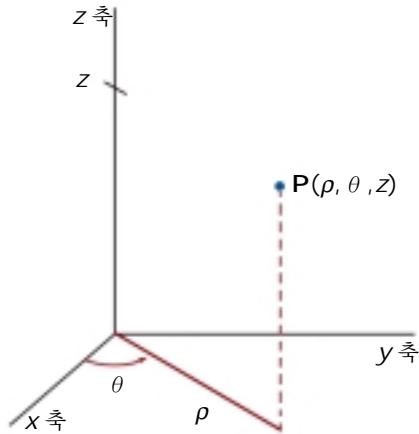
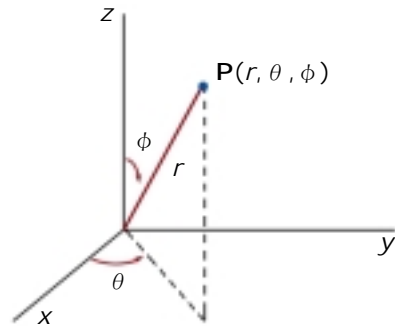
그림 부-10에서는 구면자리표계에서 공간적인 위치의 지적을 직각자리표계와의 관계속에서 보여주었다. 구면자리표를 공간에서의 극자리표라고도 한다. 상수 r면은 구, 상수 θ면은 z축을 포함하는 수직평면(원기둥자리표계에서의 θ면과 같다.), 상수 ϕ면은 자리표원점에 정점이 있는 원추이다. ϕ <90°이면 원추는 xy평면의 우에 있고 ϕ>90°이면 원추는 xy평면의 아래에 있다. 계산

$$x=r\cos\theta\sin\phi,\quad y=r\sin\theta\sin\phi,\quad z=r\cos\phi \qquad (부-5)$$

에 의해 구면자리표를 직각자리표로 변환한다.

그림 부-9. 원기둥자리표 ρ, θ, z

그림 부-10. 구면자리표 r, θ, ϕ

립체각

두 사귐선사이의 2차원각 θ(식 부-3)와 류사하게 립체각을 정의한다. 원대신에 중심위치가 **P**인 구를 고찰하자. 정점이 **P**에 있는 원추모양공간의 립체각 ω는

$$\omega=\frac{A}{r^2} \qquad (부-6)$$

와 같이 정의된다. 여기서 A는 원추와 사귀는 구면의 면적(그림 부-11), r는 구의 반경이다.

2차원의 경우와 류사하게 립체각의 무본단위를 스테라디안이라고 한다. 점주위의 전체 립체각은 구면의 전체 면적$(4\pi r^2)$을 r^2으로 나눈 4π sr이다.

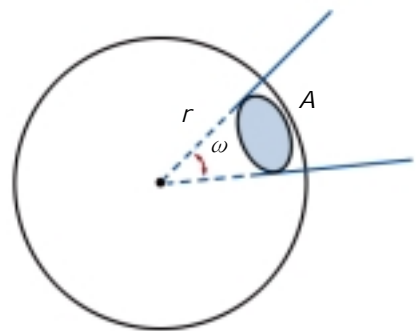



그림 부-11. 면적이 A이고 반경이 r인 구면쪼각이 마주하는 립체각 ω

부록 2. 점과 벡토르

점과 벡토르의 개념사이에는 근본적인 차이가 있다. 점은 자리표계에서 자리표값들에 의하여 지적되는 위치이며 원점으로부터의 거리는 자리표계의 선택에 관계된다. 그림 부-12에서는 두개의 자리표계에서 점의 자리표지적을 보여 주었다. 자리표계 A에서 점의 자리표는 (x, y)이다. 자리표계 B에서 같은 점의 자리표는 (0, 0)이며 자리표계 B의 원점까지의 거리는 0이다.

벡토르는 두 점의 위치차로 정의된다. 2차원벡토르(그림 부-13)에 대하여

$$\begin{aligned}\mathbf{V} &= \mathbf{P}_2 - \mathbf{P}_1 \\ &= (x_2 - x_1\ ,\ y_2 - y_1) \\ &= (V_x\ ,\ V_y)\end{aligned} \qquad (부-7)$$

이다. 여기서 직각성분(또는 직각원소) V_x와 V_y는 $\mathbf{V}$의 x와 y축에로의 사영이다. 두 점의 위치가 주어지면 임의의 자리표계에서 같은 방법으로 벡토르의 성분들을 구할수 있다.

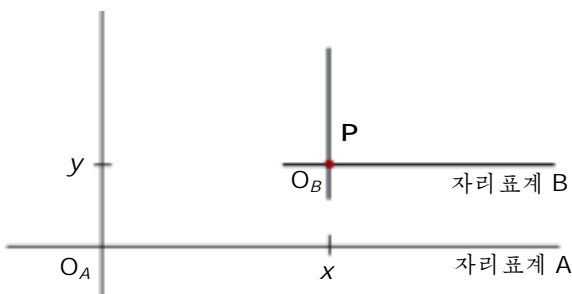



그림 부-12. 두개의 직각자리표계에서 점 **P**의 위치

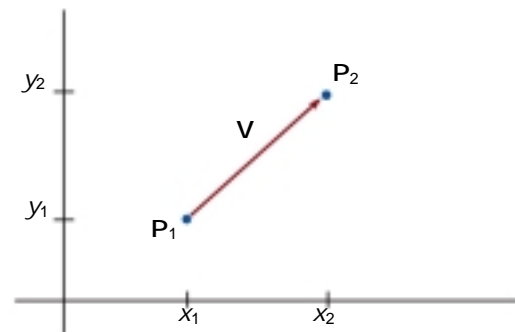



그림 부-13. 직각자리표계의 xy 평면에서 벡토르 **V**

벡토르는 크기와 방향을 가지는 방향선분으로 표현할수 있다. 그림 부-13의 2차원벡토르에 대하여 벡토르의 크기는 피다고라스정리를 리용하여

$$|\mathbf{V}| = \sqrt{V_x^2 + V_y^2} \qquad (부-8)$$

와 같이 계산된다. 2차원벡토르의 방향은 x축으로부터의 각변위에 의하여

$$\alpha = \tan^{-1}\left(\frac{V_y}{V_x}\right) \qquad (부-9)$$

와 같이 주어 질수 있다. 벡토르는 주어 진 자리표계에서 그것을 어디에 놓는가에 관계없이 일정한 크기와 방향을 가진다. 벡토르성분들의 값은 자리표표시에 따라 변하지만 벡토르의 크기는 자리표표시에 무관계하다.

3차원직각자리표계에서 벡토르의 크기는

$$|\mathbf{V}| = \sqrt{V_x^{\ 2} + V_y^{\ 2} + V_z^{\ 2}} \qquad (부-10)$$

와 같이 계산된다. 벡토르의 방향은 벡토르가 매개 자리표축과 만드는 방향각 α, β, γ에 의하여 주어 진다(그림 부-14). 방향각은 벡토르가 매개 정의 자리표축과 이루는 정의 각이다. 이 각들은

$$\cos\alpha = \frac{V_x}{|\mathbf{V}|}, \quad \cos\beta = \frac{V_y}{|\mathbf{V}|}, \quad \cos\gamma = \frac{V_z}{|\mathbf{V}|} \qquad (부-11)$$

와 같이 계산된다. 값 $\cos\alpha$, $\cos\beta$, $\cos\gamma$를 벡토르의 방향코시누스라고 한다. 실제로 $\mathbf{V}$의 방향을 주기 위하여 두개의 방향코시누스만을 지적하면 된다. 왜냐하면

$$\cos^2\alpha + \cos^2\beta + \cos^2\gamma = 1 \qquad (부-12)$$

이기때문이다.

벡토르는 크기와 방향을 가지는 량을 표현하는데 리용된다. 두가지 일반적인 실례는 힘과 속도이다(그림 부-15). 힘은 주어 진 방향에서 일정한 크기의 밀기 또는 당기기로 생각할수 있다. 속도벡토르는 물체가 일정한 방향에서 얼마나 빨리 움직이는가를 지적한다.

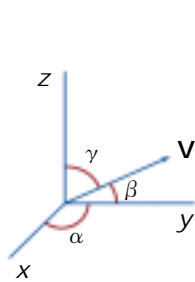



그림 부-14. 방향각 α, β, γ

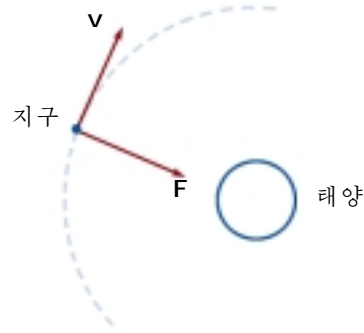



그림 부-15. 인력벡토르 **F**와 속도벡토르 **v**

벡토르더하기와 스칼라곱하기

정의에 의하여 두 벡토르의 합은 대응하는 성분들을 더하여 얻는다.

$$\mathbf{V}_1 + \mathbf{V}_2 = (V_{1x} + V_{2x}\ , V_{1y} + V_{2y}\ , V_{1z} + V_{2z}) \qquad (부-13)$$

그림 부-16에서는 기하학적인 벡토르의 더하기를 보여 주었다. 벡토르의 합은 한 벡토르의 시작위치를 다른 벡토르의 끝위치에 놓고 그림 부-16에서와 같이 합벡토르를 그어 얻는다.

벡토르와 스칼라의 더하기는 정의되지 않는다. 왜냐하면 스칼라는 항상 한개의 수값만을 가지지만 벡토르는 n차원공간에서 n개의 수값성분을 가지기때문이다. 3차원벡토르의 스칼라곱하기는

$$a\mathbf{V} = (aV_x\, , aV_y\, , aV_z) \qquad (부-14)$$

와 같이 정의된다. 실례로 스칼라파라메터 a가 값 2를 가지면 $\mathbf{V}$의 매개 성분은 2배 된다.

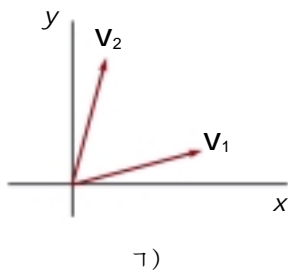
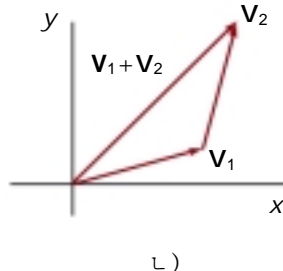
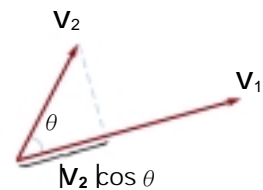



그림 부-16. 두 벡토르(ㄱ)는 한 벡토르의 끝과 다른 벡토르의 시작을 매 놓고(ㄴ) 첫번째 벡토르의 시작에서 두번째 벡토르의 끝까지 결과벡토르를 그어 기하학적으로 더할수 있다.

그림 부-17. 두 벡토르의 스칼라적은 평행성분들을 곱하여 얻는다.

두 벡토르를 곱하는데는 두가지 방법이 있다. 곱하기는 벡토르를 얻든가 또는 스칼라를 얻도록 수행될수 있다.

두 벡토르의 스칼라적

스칼라를 만드는 벡토르곱하기는

$$\mathbf{V_1}\cdot\mathbf{V_2}=|\mathbf{V}_1|\,|\mathbf{V}_2|\cos\theta,\qquad 0\le\theta\le\pi \qquad (부-15)$$

와 같이 정의된다. 여기서 θ는 두 벡토르사이의 각이다(그림 부-17). 이 적을 두 벡토르의 스칼라적 또는 점적이라고 한다. 특히 텐소르해석에서는 스칼라적론의에서 이것을 내적이라고 한다. 식 부-15는 임의의 자리표표시에서 유효하며 두 벡토르의 평행성분들의 적으로 해석할수 있다.

스칼라적의 자리표독립형식외에 이 적을 구체적인 자리표로 표시할수 있다. 직각자리표계에서 스칼라적은

$$\mathbf{V}_1\cdot\mathbf{V}_2=(V_{1x}V_{2x}+V_{1y}V_{2y}+V_{1z}V_{2z}) \qquad (부-16)$$

와 같이 계산된다. 벡토르의 자기 자체와의 스칼라적은 간단히 피다고다스정리의 다른 설명이다. 또한 두 벡토르의 스칼라적은 두 벡토르가 수직일 때에만 0이다. 스칼라적은 교환성을 가진다.

$$\mathbf{V}_1\cdot\mathbf{V}_2=\mathbf{V}_2\cdot\mathbf{V}_1 \qquad (부-17)$$

왜냐하면 이 연산은 스칼라를 만들기때문이다. 그리고 스칼라적은 벡토르의 더하기에 대하여 분배성을 가진다.

$$\mathbf{V}_1\cdot(\mathbf{V}_2+\mathbf{V}_3)=\mathbf{V}_1\cdot\mathbf{V}_2+\mathbf{V}_1\cdot\mathbf{V}_3 \qquad (부-18)$$

두 벡토르의 벡토르적

벡토르를 만드는 두 벡토르의 곱하기는

$$\mathbf{V}_1\times\mathbf{V}_2=\mathbf{u}|\mathbf{V}_1|\,|\mathbf{V}_2|\sin\theta,\qquad 0\le\theta\le\pi \qquad (부-19)$$

와 같이 정의된다. 여기서 $\mathbf{u}$는 $\mathbf{V}_1$과 $\mathbf{V}_2$에 다같이 수직인 단위벡토르(크기1)이다(그림 부-18). $\mathbf{u}$의 방향은 오른손규칙에 의하여 결정된다. 오른손 손가락들을 $\mathbf{V}_1$로부터 $\mathbf{V}_2$에로 구부려 $\mathbf{V}_1$와 $\mathbf{V}_2$에 수직인 축을 움켜 쥘 때 오른손 엄지손가락은 $\mathbf{u}$의 방향을 가리킨다. 이 적을 두 벡토르의 벡토르적 또는 승적이라고 한다. 식 부-19는 임의의 자리표표시에서 유효하다. 두 벡토르의 벡토르적은 두 벡토르가 만드는 면에 수직이고 두 벡토르에 의하여 형성되는 평행4변형의 면적과 같은 크기를 가지는 벡토르이다.

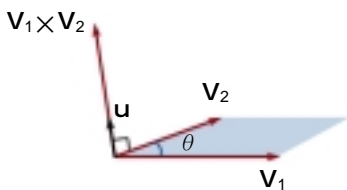



그림 부-18. 두 벡토르의 벡토르적은 방향이 본래의 두 벡토르에 수직이고 크기가 색칠된 평행4변형의 면적과 같은 벡토르이다.

벡토르적은 또한 구체적인 자리표계에서 벡토르의 성분들에 의하여 표현할수 있다. 직각자리표계에서 벡토르적의 성분들은

$$\mathbf{V}_1\times\mathbf{V}_2=(V_{1y}V_{2z}-V_{1z}V_{2y}\,,\,V_{1z}V_{2x}-V_{1x}V_{2z}\,,\,V_{1x}V_{2y}-V_{1y}V_{2x}) \qquad (부-20)$$

와 같이 계산된다. $\mathbf{u}_x$, $\mathbf{u}_y$, $\mathbf{u}_z$가 x, y, z축의 단위벡토르(크기1)라고 하면 벡토르적을 직각성분들의 행렬식표기법을 리용하여 쓸수 있다.

$$\mathbf{V}_1\times\mathbf{V}_2=\begin{vmatrix}\mathbf{u_x} & \mathbf{u_y} & \mathbf{u_z}\\ V_{1x} & V_{1y} & V_{1z}\\ V_{2x} & V_{2y} & V_{2z}\end{vmatrix} \qquad (부-21)$$

평행인 두 벡토르의 벡토르적은 령이다. 따라서 벡토르의 자기자체와의 벡토르적은 령이다. 벡토르적은 교환성을 가지지 않으며 반대교환성을 가진다.

$$\mathbf{V}_1 \times \mathbf{V}_2 = -(\mathbf{V}_2 \times \mathbf{V}_1) \qquad (부-22)$$

그리고 벡토르적은 결합성을 가지지 않는다.

$$\mathbf{V}_1 \times (\mathbf{V}_2 \times \mathbf{V}_3) \neq (\mathbf{V}_1 \times \mathbf{V}_2) \times \mathbf{V}_3 \qquad (부-23)$$

그러나 벡토르적은 벡토르의 더하기에 대하여 분배성을 가진다. 즉

$$\mathbf{V}_1 \times (\mathbf{V}_2 + \mathbf{V}_3) = (\mathbf{V}_1 \times \mathbf{V}_2) + (\mathbf{V}_1 \times \mathbf{V}_3) \qquad (부-24)$$

부록 3. 토대벡토르와 계량텐소르

임의의 자리표계에서 자리표축들은 매 축에 하나씩인 벡토르들의 모임에 의하여 지적할수 있다(그림 부-19). 매개 자리표축벡토르는 축의 임의의 점에서 그 축의 방향을 준다. 이 벡토르들은 1차독립인 벡토르들의 모임을 형성한다. 즉 매개 자리표축벡토르는 다른것들의 1차결합으로 표시할수 없다. 또한 그 공간의 다른 임의의 벡토르는 자리표축벡토르들의 1차결합으로 표시할수 있다. 자리표축벡토르들의 모임을 공간의 토대 또는 토대벡토르들의 모임이라고 한다. 일반적으로 공간이 벡토르공간이라고 할 때 토대는 공간의 임의의 벡토르를 토대벡토르들의 1차결합으로 표시하는 최소한의 벡토르들을 포함한다.

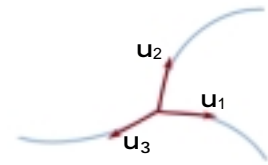



그림 부-19. 곡선자리표축벡토르

표준직교토대

토대벡토르들은 자주 매개 벡토르가 크기 1을 가지도록 표준화된다. 이 경우 단위토대벡토르들의 모임을 표준토대라고 한다. 또한 직각자리표계와 기타 일반적으로 리용되는 자리표계들에서 자리표축들은 서로 수직이며 이때 토대벡토르들의 모임을 직교토대라고 한다. 토대벡토르들이 모두 단위벡토르이면 다음의 조건

$$\begin{aligned} \mathbf{u}_k \cdot \mathbf{u}_k &= 1, \quad 임의의\ k \\ \mathbf{u}_j \cdot \mathbf{u}_k &= 0, \quad 임의의\ j \neq k \end{aligned} \qquad (부-25)$$

을 만족시키는 토대를 표준직교토대라고 한다. 가장 일반적으로 리용되는 자리표계는 직각자리표계이지만 빗각자리표계는 상대성리론과 일정한 자료모임들의 가시화 등에 리용된다.

2차원직각자리표계에서 표준직교토대는

$$\mathbf{u}_x = (1,0), \qquad \mathbf{u}_y = (0,1) \qquad (부-26)$$

이다. 그리고 3차원직각자리표계에서 표준직교토대는

$$\mathbf{u}_x = (1,0,0), \quad \mathbf{u}_y = (0,1,0), \quad \mathbf{u}_z = (0,0,1) \qquad (부-27)$$

이다.

계량텐소르

텐소르는 벡토르개념의 일반화이다. 구체적으로 텐소르는 텐소르차수와 공간의 차원수에 의존하여 한 자리표표현으로부터 다른것으로 변환될 때 일정한 변환성질을 만족시키는 몇개의 성분들을 가지는 량이다. 직각자리표계에 대한 변환성질은 간단하다. 형식적으로 벡토르는 차수가 1인 텐소르이

며 스칼라는 차수가 0인 텐소르이다. 벡토르의 성분들은 하나의 첨수에 의해 지적되며 한편 스칼라는 언제나 한개의 값을 가지며 따라서 첨수가 없다는데 주의하자. 이리하여 차수가 2인 텐소르는 2개의 첨수를 가지며 3차원공간에서 차수가 2인 텐소르는 9개의 성분(매개 첨수에 대하여 3개의 값)을 가진다.

임의의 일반적인 자리표계(곡선)에서 그 공간에 대한 계량텐소르의 원소(또는 결수)들은

$$g_{jk} = \mathbf{u}_j \cdot \mathbf{u}_k \qquad \text{(부-28)}$$

와 같이 정의된다. 이리하여 계량텐소르는 차수가 2이며 대칭 즉 $g_{jk}=g_{kj}$이다. 계량텐소르는 여러가지 쓸모 있는 성질들을 가진다. 계량텐소르의 원소들은 (1) 그 공간에서 두 점사이 거리, (2) 다른 공간에로의 변환식, (3) 그 공간에서 여러가지 미분벡토르연산자(그라디엔트, 발산, 회전 등)의 성분들을 결정하는데 리용할수 있다.

직교공간에서

$$g_{jk} = 0, \qquad \text{임의의} \quad j \neq k \qquad \text{(부-29)}$$

이다. 그리고 직각자리표계에서(단위토대벡토르들을 가정하면)

$$g_{jk} = \begin{cases} 1, & j = k \text{ 일 때} \\ 0, & \text{그밖의 경우} \end{cases} \qquad \text{(부-30)}$$

이다.

극자리표계의 단위토대벡토르들은 직교토대벡토르들에 의하여

$$\mathbf{u}_r = \mathbf{u}_x \cos\theta + \mathbf{u}_y \sin\theta, \qquad \mathbf{u}_\theta = -\mathbf{u}_x r \sin\theta + \mathbf{u}_y r \cos\theta \qquad \text{(부-31)}$$

와 같이 표현할수 있다. 이 식들을 식 부-28에 대입하면 계량텐소르의 원소들을 얻으며 그것은 행렬형식으로 쓸수 있다.

$$\mathbf{g} = \begin{bmatrix} 1 & 0 \\ 0 & r^2 \end{bmatrix} \qquad \text{(부-32)}$$

원기둥자리표계의 토대벡토르들은

$$\mathbf{u}_\rho = \mathbf{u}_x \cos\theta + \mathbf{u}_y \sin\theta, \quad \mathbf{u}_\theta = -\mathbf{u}_x \rho \sin\theta + \mathbf{u}_y \rho \cos\theta, \quad \mathbf{u}_z \qquad \text{(부-33)}$$

이다. 그리고 원기둥자리표계에서 계량텐소르의 행렬표현은

$$\mathbf{g} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \rho & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(부-34)}$$

이다.

구면자리표계의 토대벡토르들은

$$\begin{aligned} \mathbf{u}_r &= \mathbf{u}_x \cos\theta \sin\phi + \mathbf{u}_y \sin\theta \sin\phi + \mathbf{u}_z \cos\phi \\ \mathbf{u}_\theta &= -\mathbf{u}_x r \sin\theta \sin\phi + \mathbf{u}_y r \cos\theta \sin\phi \\ \mathbf{u}_\phi &= \mathbf{u}_x r \cos\theta \cos\phi + \mathbf{u}_y r \sin\theta \cos\phi - \mathbf{u}_z r \sin\phi \end{aligned} \qquad \text{(부-35)}$$

와 같이 쓸수 있다. 구면자리표계에서 계량텐소르의 행렬표현은

$$\mathbf{g} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & r^2 \sin^2\phi & 0 \\ 0 & 0 & r^2 \end{bmatrix} \qquad \text{(부-36)}$$

이다.

부록 4. 행렬

행렬은 행렬의 원소라고 부르는 량(수, 함수, 수식)들의 직4각형배렬이다. 행렬의 몇가지 실례는

$$\begin{bmatrix} 3.60 & -0.01 & 2.00 \\ -5.46 & 0.00 & 1.63 \end{bmatrix}, \quad \begin{bmatrix} e^x & x \\ e^{2x} & x^2 \end{bmatrix}, \quad \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}, \quad \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad (부-37)$$

이다. 행렬은 행과 렬의 개수에 따라 식별한다. 우의 실례에서 행렬은 왼쪽에서 오른쪽으로 가면서 2×3, 2×2, 1×3, 3×1이다. 두번째 실례에서와 같이 행과 렬의 개수가 같은 행렬을 바른행렬이라고 한다.

일반적으로 $m \times n$행렬을

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \qquad (부-38)$$

와 같이 쓸수 있다. 여기서 a_{jk}는 행렬 **A**의 원소를 표현한다. 원소의 첫번째 첨수는 행의 번호이며 두번째 첨수는 렬의 번호이다.

한개의 행 또는 한개의 렬만을 가지는 행렬은 벡토르를 표현한다. 식 부-37에서 마지막두개의 행렬실례는 각각 행벡토르와 렬벡토르이다. 일반적으로 행렬은 행벡토르 또는 렬벡토르들의 모임으로 볼수 있다. 여러가지 연산들이 행렬형식으로 표현될 때 표준적인 수학표기는 벡토르를 렬벡토르로 표현하는것이다. 이 표기에 따라 직각자리표계에서 3차원벡토르의 행렬표현을

$$\mathbf{V} = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \qquad (부-39)$$

와 같이 쓴다. 이 행렬표현을 점과 벡토르에 대하여 다같이 리용하겠다. 그러나 그것들사이의 차이를 기억하여야 한다. 점을 자리표계의 원점으로부터 시작되는 벡토르로 고찰하는것은 편리하다. 그러나 점은 한 자리표계로부터 다른데로 변환할 때 불변인 벡토르의 성질을 가지지 않는다. 또한 일반적으로 벡토르더하기, 스칼라적, 벡토르적과 같은 벡토르연산들을 점에 적용할수 없다.

스칼라곱하기와 행렬더하기

행렬 **A**에 스칼라값 s를 곱하기 위하여 매개 원소 a_{jk}에 스칼라를 곱한다. 실례로

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

이면

$$3\mathbf{A} = \begin{bmatrix} 3 & 6 & 9 \\ 12 & 15 & 18 \end{bmatrix}$$

이다.

행렬더하기는 동일한 개수의 행 m과 동일한 개수의 렬 n을 가지는 행렬들에 대해서만 정의된다. 임의의 두개의 $m \times n$행렬에서 합은 대응하는 원소들을 더하여 얻는다. 실례로

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} + \begin{bmatrix} 0 & 1.5 & 0.2 \\ -6 & 1.1 & -10 \end{bmatrix} = \begin{bmatrix} 1 & 3.5 & 3.2 \\ -2 & 6.1 & -4 \end{bmatrix}$$

이다.

행렬곱하기

두 행렬의 곱하기는 벡토르의 스칼라적의 일반화로 정의된다. $m \times n$행렬 **A**와 $p \times q$행렬 **B**는 **A**의 렬의 개수와 **B**의 행의 개수가 같으면(즉 $n=p$) 적행렬 **AB**를 만들기 위하여 곱할수 있다. 적행렬은 **A**의 행벡토르의 원소들과 **B**의 렬벡토르의 대응하는 원소들의 적의 합을 만들어 얻는다. 그리하여 다음의 곱하기

$$\mathbf{C} = \mathbf{AB} \qquad (부-40)$$

에서 원소들이

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \qquad (부-41)$$

와 같이 계산되는 $m \times q$행렬 **C**를 얻는다.

다음의 실례에서 3×2적행렬을 만들기 위하여 3×2행렬을 2×2행렬의 왼쪽에 곱한다.

$$\begin{bmatrix} 0 & -1 \\ 5 & 7 \\ -2 & 8 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 0\cdot1+(-1)\cdot3 & 0\cdot2+(-1)\cdot4 \\ 5\cdot1+7\cdot3 & 5\cdot2+7\cdot4 \\ -2\cdot1+8\cdot3 & -2\cdot2+8\cdot4 \end{bmatrix} = \begin{bmatrix} -3 & -4 \\ 26 & 38 \\ 22 & 28 \end{bmatrix}$$

첫번째 벡토르를 행벡토르로 표현하고 두번째 벡토르를 렬벡토르로 표현하여 행렬곱하기하면 두 벡토르의 스칼라적과 같은 결과를 만든다.

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 32 \end{bmatrix}$$

이 행렬곱하기는 한개의 원소를 가지는 행렬(1×1행렬)을 만든다. 행렬들을 반대순서로 곱하면 3×3행렬을 얻는다.

$$\begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 4 & 8 & 12 \\ 5 & 10 & 15 \\ 6 & 12 & 18 \end{bmatrix}$$

앞의 두 행렬곱하기가 설명하는바와 같이 행렬의 곱하기는 일반적으로 교환성을 가지지 않는다. 즉

$$\mathbf{AB} \neq \mathbf{BA} \qquad (부-42)$$

그러나 행렬의 곱하기는 행렬의 더하기에 대하여 분배성을 가진다.

$$\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC} \qquad (부-43)$$

전위행렬

전위행렬 A^T는 행과 렬을 교체하여 얻는다. 실례로

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}, \qquad \begin{bmatrix} a & b & c \end{bmatrix}^T = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \qquad (부-44)$$

행렬곱하기에 대하여 전위는

$$(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T \qquad (부-45)$$

행렬식

바른행렬에 대하여 행렬식이라고 하는 한개의 수를 만들기 위하여 행렬의 원소들을 결합할수 있다. 행렬식은 재귀적으로 정의된다. 2×2행렬에 대하여 2차행렬식은

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21} \qquad (부-46)$$

로 정의된다. 보다 높은 차수의 행렬식은 보다 낮은 차수의 행렬식에 의하여 계산한다. 3차 또는 그 이상의 행렬식을 계산하기 위하여 $n\times n$행렬의 임의의 렬 k를 선택하고 행렬식을

$$\mathrm{det}\mathbf{A} = \sum_{j=1}^{n} (-1)^{j+k} a_{jk} \mathrm{det}\mathbf{A}_{jk} \qquad (부-47)$$

와 같이 계산할수 있다. 여기서 det$\mathbf{A}_{jk}$는 $\mathbf{A}$에서 j번째 행과 k번째 렬을 지워 얻어지는 $(n-1)\times(n-1)$ 부분행렬의 행렬식이다. 또 다르게 임의의 행 j를 선택하고 행렬식을

$$\mathrm{det}\mathbf{A} = \sum_{k=1}^{n} (-1)^{j+k} a_{jk} \,\mathrm{det}\,\mathbf{A}_{jk} \qquad (부-48)$$

와 같이 계산할수 있다.

큰 행렬(말하자면 $n>4$)에 대한 행렬식계산은 수값방법들을 리용하여 보다 효률적으로 진행할수 있다. 행렬식을 계산하는 한가지 방법은 행렬을 두개의 인자로 분해하는것이다. 즉 $\mathbf{A}=\mathbf{LU}$, 여기서 행렬 $\mathbf{L}$의 대각선우의 모든 원소들과 행렬 $\mathbf{U}$의 대각선아래의 모든 원소들은 0이다. 그러면 $\mathbf{L}$과 $\mathbf{U}$에 대하여 다같이 대각선의 적을 계산하고 det$\mathbf{A}$는 이 두 적을 함께 곱하여 얻는다. 이 방법은 행렬식의 다음의 성질에 기초한다.

$$\mathrm{det}(\mathbf{AB}) = (\mathrm{det}\mathbf{A})(\mathrm{det}\mathbf{B}) \qquad (부-49)$$

행렬식계산의 다른 방법은 가우스소거절차(부록 9)에 기초한다.

역행렬

바른행렬에 대하여 행렬의 행렬식이 0이 아닐 때에만 역행렬을 얻을수 있다. 역행렬이 존재하는 행렬을 불퇴화행렬이라고 하며 그렇지 않은 행렬을 퇴화행렬이라고 한다. 대부분의 실천응용들에서 행렬이 물리적인 조작을 표현할 때 역행렬이 존재한다고 볼수 있다.

$n\times n$바른행렬 $\mathbf{A}$의 역행렬은 $\mathbf{A}^{-1}$로 표시하며

$$\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I} \qquad (부-50)$$

이다. 여기서 $\mathbf{I}$는 단위행렬이다. $\mathbf{I}$의 모든 대각선원소들은 값 1을 가지며 다른 모든 원소들은(비대각선) 0이다.

역행렬 $\mathbf{A}^{-1}$의 원소들은 $\mathbf{A}$의 원소들로부터

$$a_{jk}^{-1} = \frac{(-1)^{j+k}\,\mathrm{det}\,\mathbf{A}_{kj}}{\mathrm{det}\,\mathbf{A}} \qquad (부-51)$$

와 같이 계산할수 있다. 여기서 a_{jk}^{-1}은 $\mathbf{A}^{-1}$의 j번째 행 및 k번째 렬의 원소이며 $\mathbf{A}_{kj}$는 행렬 $\mathbf{A}$의 k번째 행과 j번째 렬을 지워 얻어 지는 $(n-1)\times(n-1)$부분행렬이다. n의 값이 큰 행렬식과 역행렬의 원소들을 계산하는데 수값방법들을 리용할수 있다.

부록 5. 복소수

정의에 의하여 복소수 z는 실수들의 순서쌍이다.

$$z = (x, y) \qquad (부-52)$$

여기서 x는 z의 실수부, y는 z의 허수부라고 한다. 복소수의 실수부와 허수부는

$$x = \mathrm{Re}(z), \qquad y = \mathrm{Im}(z) \qquad (부-53)$$

로 나타낸다. 복소수는 기하학적으로 그림 부-20에서와 같이 복소수평면에서 표현된다.

복소수는

$$x^2 + 1 = 0, \qquad x^2 - 2x + 5 = 0$$

과 같은 방정식들의 풀이로부터 생긴다. 그것들은 실수풀이가 없다. 그러므로 복소수와 복소수연산은 이런 방정식들의 풀이를 주는 실수의 확장으로 된다.

복소수의 더하기, 덜기, 스칼라곱하기는 2차원벡토르와 같은 규칙을 리용하여 수행한다. 복소수의 곱하기는

$$(x_1, y_1)(x_2, y_2) = (x_1x_2 - y_1y_2, x_1y_2 + x_2y_1) \qquad (부-54)$$

와 같이 정의된다. 복소수에 대한 이 정의는 허수부가 0일 때 실수곱하기와 같은 결과를 준다.

$$(x_1, 0)(x_2, 0) = (x_1x_2, 0)$$

그러므로 실수를 복소수형식으로

$$x = (x, 0)$$

과 같이 쓸수 있다. 류사하게 순허수는 0의 실수부를 가진다. 즉 $(0, y)$

복소수 $(0, 1)$을 허수단위라고 하며

$$i = (0, 1) \qquad (부-55)$$

로 표시한다. 전기기사들은 허수단위에 대하여 기호 j를 리용한다. 왜냐하면 기호 i는 전류를 표현하는데 리용되기때문이다. 복소수곱하기규칙으로부터

$$i^2 = (0, 1)(0, 1) = (-1, 0)$$

이다. 따라서 i^2은 실수 -1이며

$$i = \sqrt{-1} \qquad (부-56)$$

이다.

복소수곱하기규칙을 리용하여 임의의 순허수는

$$iy = (0, 1)(0, y) = (0, y)$$

의 형식으로 쓸수 있다. 또한 더하기규칙에 의하여 임의의 복소수는 합

$$z = (x, 0) + (0, y)$$

으로 쓸수 있다. 따라서 복소수에 대한 다른 표현은

$$z = x + iy \qquad (부-57)$$

이다. 이것은 실천응용들에서 리용되는 보통의 형식이다.

복소수와 관련되는 다른 개념은 공액복소수이다.

$$\bar{z} = x - iy \qquad (부-58)$$

복소수의 절대값은

$$|z| = \sqrt{z\bar{z}} = \sqrt{x^2 + y^2} \qquad (\text{부}-59)$$

로 정의된다. 그것은 복소수를 표현하는 벡토르의 길이(즉 복소수평면의 원점으로부터 점 z까지의 거리)를 준다. 두 복소수의 나누기에 대하여 실수부와 허수부는

$$\begin{aligned}\frac{z_1}{z_2} &= \frac{z_1\bar{z_2}}{z_2\bar{z_2}} \\ &= \frac{(x_1, y_1)(x_2, -y_2)}{x_2^2 + y_2^2} \\ &= \left(\frac{x_1x_2 + y_1y_2}{x_2^2 + y_2^2}, \frac{x_2y_1 - x_1y_2}{x_2^2 + y_2^2}\right)\end{aligned} \qquad (\text{부}-60)$$

와 같이 얻어 진다.

복소수에 대한 특히 쓸모 있는 표현은 실수부와 허수부를 극자리표에 의하여 표현하는것이다(그림 부-21).

$$z = r(\cos\theta + i\sin\theta) \qquad (\text{부}-61)$$

z의 극형식은 또한

$$z = re^{i\theta} \qquad (\text{부}-62)$$

와 같이 쓸수 있다. 여기서 e는 자연로그의 밑수 (e=2.718281828...)이며

$$e^{i\theta} = \cos\theta + i\sin\theta \qquad (\text{부}-63)$$

이다. 이것은 오일레르공식이다. 복소수의 곱하기와 나누기는

$$z_1z_2 = r_1r_2e^{i(\theta_1+\theta_2)}, \qquad \frac{z_1}{z_2} = r_1/r_2\, e^{i(\theta_1-\theta_2)}$$

와 같이 쉽게 얻어 진다. 그리고 복소수의 n차뿌리는

$$\sqrt[n]{z} = \sqrt[n]{r}\left[\cos\left(\frac{\theta + 2k\pi}{n}\right) + i\sin\left(\frac{\theta + 2k\pi}{n}\right)\right], \quad k = 0, 1, 2, \ldots, n-1 \qquad (\text{부}-64)$$

와 같이 계산된다. n차뿌리들은 중심이 복소수평면의 원점에 있는 반경이 $\sqrt[n]{r}$ 인 원에 놓이며 바른n각형의 정점들을 형성한다.

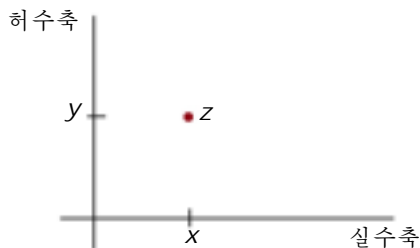



그림 부-20. 복소수평면에서 점 z 의 위치

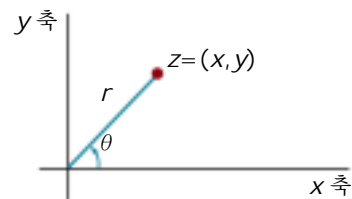



그림 부-21. 복소수 z 의 극자리표위치

부록 6. 4원수

복소수개념은 4원수에 의하여 더 높은 차원에로 확장된다. 4원수(quaternion)는 한개의 실수부와 세개의 허수부를 가지는 수이며

$$q = s + ia + jb + kc \qquad (부-65)$$

와 같이 쓴다. 여기서 허수항에서의 결수 a, b, c는 실수이며 파라메터 s는 스칼라부라고 하는 실수이다. 파라메터 i, j, k는 성질

$$i^2 = j^2 = k^2 = -1, \qquad ij = -ji = k \qquad (부-66)$$

에 의하여 정의된다. 이 성질들로부터

$$jk = -kj = i, \quad ki = -ik = j \qquad (부-67)$$

이 나온다.

스칼라곱하기는 벡토르 및 복소수에 대한 대응하는 연산과 류사하게 정의된다. 즉 4원수의 네개의 매 성분에 스칼라값을 곱한다. 류사하게 4원수의 더하기는

$$q_1 + q_2 = (s_1 + s_2) + i(a_1 + a_2) + j(b_1 + b_2) + k(c_1 + c_2) \qquad (부-68)$$

와 같이 정의된다. 두 4원수의 곱하기는 식 부-66과 부-67에서의 연산들을 리용하여 수행한다.

4원수에 대한 순서쌍표기는 또한 복소수표기와 류사하게 만들어 진다.

$$q = (s, \mathbf{v}) \qquad (부-69)$$

여기서 $\mathbf{v}$는 벡토르 (a,b,c)이다. 이 표기에서 4원수더하기는

$$q_1 + q_2 = (s_1 + s_2\ ,\ \ \mathbf{v}_1 + \mathbf{v}_2) \qquad (부-70)$$

로 표현된다. 그러면 4원수곱하기는 벡토르의 스칼라적과 벡토르적에 의하여

$$q_1 q_2 = (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2\ ,\ \ s_1 \mathbf{v}_2 + s_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2) \qquad (부-71)$$

와 같이 표현할수 있다.

복소수연산의 확장으로서 4원수의 두제곱크기는 벡토르의 스칼라적을 리용하여

$$|q|^2 = s^2 + \mathbf{v} \cdot \mathbf{v} \qquad (부-72)$$

와 같이 정의된다. 그리고 4원수의 역수는

$$q^{-1} = \frac{1}{|q|^2}(s\ ,\ \ -\mathbf{v}) \qquad (부-73)$$

이며 따라서

$$qq^{-1} = q^{-1}q = (1,\ 0)$$

부록 7. 비보조변수표현

물체를 자리표계의 자리표들에 의하여 직접 표현할 때의 표현을 비보조변수표현이라고 한다. 실례로 곡면은 다음의 함수

$$f(x,\ y,\ z) = 0 \text{ 또는 } \quad z = f(x,\ y) \qquad (부-74)$$

에 의하여 표현할수 있다. 식 부-74에서 첫번째 형식은 곡면에 대한 음함수적표현이며 두번째 형식은 x, y가 독립변수이고 z가 종속변수인 양함수적표현이다.

류사하게 두 곡면함수의 사귐인 3차원곡선은 비보조변수형식으로도 표현할수 있고 함수들의 쌍에 의해서도 표현할수 있다.

$$y = f(x), \qquad z = g(x) \qquad (부-75)$$

여기서 자리표 x는 독립변수이다. 종속변수 y 및 z의 값들은 곡선의 한 끝점으로부터 다른 끝점까지 x의 값을 변화시키는데 따라 식 부-75로부터 결정된다.

비보조변수표현은 물체를 주어 진 자리표계에서 표현하는데 쓸모 있지만 도형처리알고리듬들에 리용할 때에는 약간의 결함이 있다. 원활한 곡선을 그리자면 $f(x)$ 또는 $g(x)$의 1계도함수(경사도)의 절대값이 1보다 클 때마다 독립변수를 바꾸어야 한다. 이것은 매점에서 도함수값(무한대일수도 있다.)을 계속 검사하여야 한다는것을 의미한다. 또한 식 부-75는 다값함수를 표현하는데 불편하다. 실례로 중심이 xy평면의 원점에 있는 원의 음함수적인 방정식은

$$x^2 + y^2 = r^2$$

이며 y에 대한 양함수적인 표현은 다값함수

$$y = \pm\sqrt{r^2 - x^2}$$

이다. 일반적으로 도형처리알고리듬들에서 물체를 표현하는데 보다 편리한 표현은 보조변수방정식에 의한 표현이다.

부록 8. 보조변수표현

유클리드곡선은 1차원물체이며 3차원곡선경로를 따르는 위치들은 한개의 파라메터 u에 의하여 표현할수 있다. 즉 3개의 매 직각자리표를 파라메터 u에 의하여 표현할수 있으며 곡선의 점은 다음의 점벡토르함수

$$\mathbf{P}(u) = (x(u),\ y(u),\ z(u)) \qquad (부-76)$$

에 의하여 표현할수 있다. 파라메터 u가 단위구간 0～1에서 정의되도록 자리표방정식을 설정할수 있다. 실례로 중심이 자리표원점에 있는 xy평면의 원은 보조변수형식으로

$$x(u) = r\cos(2\pi u), \quad y(u) = r\sin(2\pi u), \quad z(u) = 0, \quad 0 \le u \le 1 \qquad (부-77)$$

와 같이 정의할수 있다. 다른 보조변수형식들을 리용하여 원과 원호를 표현할수도 있다.

유클리드곡면(또는 평면)은 2차원물체이며 면우의 위치들은 2개의 파라메터 u와 v에 의하여 표현할수 있다. 면에서의 자리표위치는 보조변수벡토르함수

$$\mathbf{P}(u,v) = (x(u,v),\ \ y(u,v),\ \ z(u,v)) \qquad (부-78)$$

에 의하여 표현된다. 여기서 x, y, z의 직각자리표값은 파라메터 u와 v의 함수로 표현된다. 곡선에서와 마찬가지로 파라메터 u와 v가 0～1사이 범위에서 정의되도록 보조변수표현을 정돈할수 있다. 실례로 중심이 자리표원점에 있는 구면은 방정식

$$\begin{aligned} x(u,v) &= r\sin(\pi u)\cos(2\pi v) \\ y(u,v) &= r\sin(\pi u)\sin(2\pi v) \qquad (부-79) \\ z(u,v) &= r\cos(\pi u) \end{aligned}$$

에 의하여 표현할수 있다. 여기서 r는 구의 반경이다. 파라메터 u는 구면의 상수위도선을 표현하며 파라메터 v는 상수경도선을 표현한다. 이 파라메터들중 하나를 고정하고 다른것을 0～1사이 범위의 부분구간에서 변화시켜 임의의 부분구면에 대한 위도선과 경도선을 그릴수 있다(그림 부-22).

그림 부-22. 식 부-79에서 상수 u 및 상수 v선들에 의하여 표현되는 부분구면

부록 9. 수값방법

콤퓨터도형처리알고리듬들에서는 선형방정식, 비선형방정식, 적분방정식 등을 풀어야 한다. 또한 자료점들의 불련속모임을 가시화하기 위하여 자료모임의 점들을 곡선 또는 곡면함수로 근사화하여 현시하는것이 편리하다. 이 절에서는 여러가지 수값문제들을 푸는 일반적인 알고리듬들을 간단히 개괄한다.

련립1차방정식풀기

변수 x_k, k=1, 2, ... , n에 대한 련립1차방정식을

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \vdots \quad & \quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n \end{aligned} \qquad (부-80)$$

와 같이 쓸수 있다. 여기서 파라메터 a_{jk}와 b_j의 값들은 알고 있다. 련립1차방정식은 행렬형식으로 표현할수 있다.

$$\mathbf{AX} = \mathbf{B} \qquad (부-81)$$

여기서 **A**는 원소들이 결수 a_{jk}인 $n\times n$바른행렬, **X**는 x_k값들의 렬벡토르, **B**는 b_j값들의 렬벡토르이다. 련립1차방정식의 풀이는 행렬형식으로

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{B} \qquad (부-82)$$

와 같이 표현할수 있다. 풀이는 결수행렬 **A**의 역행렬에 관계된다. 그러므로 련립1차방정식은 **A**가 불퇴화행렬 즉 그의 행렬식이 0이 아닐 때에만 풀수 있다.

련립1차방정식을 푸는 한가지 방법은 크라메르규칙이다.

$$x_k = \frac{\det \mathbf{A}_k}{\det \mathbf{A}} \qquad (부-83)$$

여기서 $\mathbf{A}_k$는 k번째 렬을 **B**의 원소들로 교체한 행렬 **A**이다. 이 방법은 변수들이 몇개 안되는 문제에 적합하다. 3∼4개이상의 변수들에 대하여 이 방법은 매개 행렬식을 계산하는데 대단히 많은 곱하기연산을 진행해야 하므로 극히 비효률적이다. $n\times n$행렬식을 하나 계산하는데 n!개 이상의 곱하기연산이 필요하다.

련립1차방정식은 가우스소거법의 변종들을 리용하여 보다 효률적으로 풀수 있다. 가우스소거법의 기본사상은 다음의 련립2원1차방정식으로 설명할수 있다.

$$x_1 + 2x_2 = -4$$
$$3x_1 + 4x_2 = 1 \qquad (부-84)$$

이 련립방정식을 풀기 위하여 첫번째 방정식에 -3을 곱하고 x_1항을 소거하기 위하여 두번째 방정식을 더할수 있다. 그러면 방정식

$$-2x_2 = 13$$

이 나오며 그것은 풀이 x_2=-13/2를 가진다. 이것은 다음에 x_1의 풀이를 얻기 위하여 본래의 방정식들중 하나에 대입될수 있으며 그것은 9이다. 소거 및 후진대입을 수행하기 위한 효률적인 알고리듬들이 창안되였다.

가우스소거법은 때때로 반올림오차가 크며 정확한 풀이를 얻는것이 불가능할수 있다. 이런 경우 가우스-자이델방법을 리용하여 풀이를 얻을수 있다. 변수 x_k의 초기추정값들로부터 시작하여 축차근사값들사이의 차가 작을 때까지 반복적으로 축차근사를 계산한다. 매 반복에서 변수의 근사값들은

$$x_1 = \frac{b_1 - a_{12}x_2 - a_{13}x_3 - \cdots - a_{1n}x_n}{a_{11}}$$
$$x_2 = \frac{b_2 - a_{21}x_1 - a_{23}x_3 - \cdots - a_{2n}x_n}{a_{22}} \qquad (부-85)$$
$$\vdots$$

와 같이 계산한다. 매 대각선원소의 절대값이 그 행의 다른 원소들의 절대값의 합보다 크도록 행렬을 재정돈할수 있으면 가우스-자이델방법은 풀이에로 수렴한다.

비선형방정식의 풀이구하기

함수 $f(x)$의 풀이는 식 $f(x)$=0을 만족시키는 x의 값이다. 비선형방정식의 풀이를 구하는 가장 일반적인 방법들중의 하나는 뉴톤-라프손알고리듬이다. 이 알고리듬은 그림 부-23에서 보여 준바와 같이 반복의 매 걸음에서 함수 $f(x)$를 직선으로 근사하는 반복절차이다. 풀이의 초기추정값 x_0으로부터 시작하여 풀이의 다음번 근사는 x_0에서의 접선이 x축과 사귀는 곳을 결정하여 x_1로 계산한다. x_0에서 곡선의 경사도(1계도함수)는

$$\frac{df}{dx} = \frac{f(x_0)}{x_0 - x_1} \qquad (부-86)$$

이다. 그리하여 풀이의 다음번 근사는

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \qquad (부-87)$$

이다. 이 절차는 축차근사들사이의 차가 충분히 작을 때까지 매번 계산된 근사를 리용하여 반복된다.

뉴톤-라프손알고리듬이 풀이에 수렴하면 그것은 다른 풀이구하기방법보다 더 빨리 수렴할것이다. 그러나 그것은 항상 수렴하는것이 아니다. 실례로 이 방법은 반복의 어느때에 도함수 $f'(x)$가 0이면 실패한다. 또한 곡선의 진동에 따라 축차근사는 풀이위치로부터 발산할수 있다. 뉴톤-라프손알고리듬은 복소변수함수 $f(z)$와 실수 및 복소수의 련립비선형함수들에도 적용할수 있다.

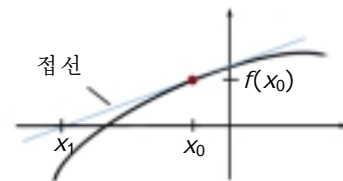



그림 부-23. 초기값 x_0에서 접선으로 곡선을 근사

속도는 뜨지만 반드시 수렴되는 다른 방법은 반분법이다.

여기서는 먼저 풀이를 포함하는 x구간을 결정하며 다음에 풀이에로 접근하기 위하여 2분탐색절차를 적용한다. 먼저 풀이가 구간의 아래 또는 웃절반에 있는가를 결정하기 위하여 구간의 중점을 조사한다. 이 절차는 축차중점위치들사이의 차가 미리 설정된 값보다 작을 때까지 매 부분구간에 대하여 계속 반복된다. 매 부분구간을 2등분하는대신에 축차 x위치들을 보간하여 계산속도를 높일수 있다(false위치방법).

적분계산

적분하는것은 합하는 과정이다. 한변수함수 $f(x)$의 적분은 그림 부-24에서 보여 주는바와 같이 곡선아래의 면적이다.

$f(x)$의 적분은 다음의 더하기

$$\int_a^b f(x)dx \approx \sum_{k=1}^{n} f_k(x)\Delta x_k \qquad (부-88)$$

에 의하여 수값근사할수 있다. 여기서 $f_k(x)$는 구간 k에서 $f(x)$의 근사이다. 실례로 매 부분구간에서 곡선을 상수로 근사시키고 결과의 직4각형들의 면적을 합할수 있다(그림 부-25). a부터 b까지의 구간을 세밀하게 분할하면 할수록 근사는 점점 더 좋아 진다. 실제로 부분구간들이 아주 작으면(점정도로까지) 직4각형면적들의 값에서 반올림오차가 거의 없을수 있다.

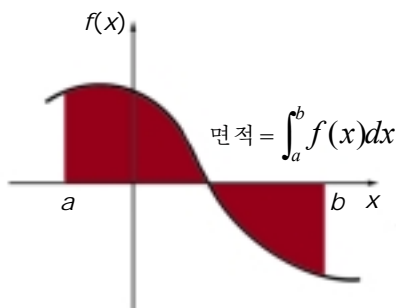



그림 부-24. $f(x)$의 적분은 a 부터 b 까지의 구간에서 함수와 x 축사이의 면적과 같다.

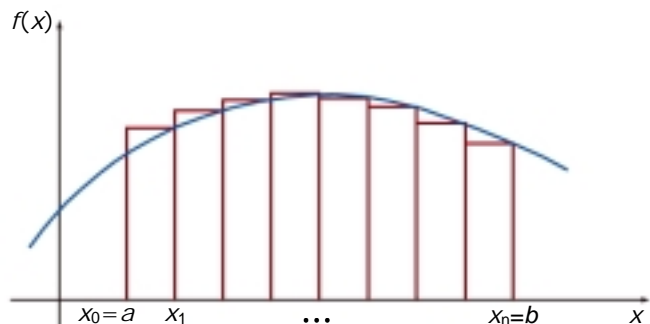



그림 부-25. 적분을 작은 직4각형면적들의 합으로 근사

일반적으로 매 부분구간에서 함수에 대한 다항식근사는 직4각형방법보다 더 좋은 결과를 준다. 선형근사를 리용하면 사다리형부분구역을 얻으며 이 근사방법을 사다리형규칙이라고 한다. 매 부분구간에서 함수를 근사시키는데 2차다항식(포물선)을 리용할 때의 방법을 심프손규칙이라고 하며 적분근사는

$$\int_a^b f(x)dx \approx \frac{\Delta x}{3}\left[f(a)+f(b)+4\sum_{홀수\ k=1}^{n-1} f(x_k)+2\sum_{짝수\ k=2}^{n-2} f(x_k)\right] \qquad (부-89)$$

이다. 여기서 a부터 b까지의 구간은 등너비구간

$$\Delta x = \frac{b-a}{n} \qquad (부-90)$$

들로 나뉘여 지며 n은 2의 배수이고

$$x_0 = a, \quad x_k = x_{k-1} + \Delta x, \quad k = 1, 2, \dots, n$$

고주파진동을 가지는 함수(그림 부-26)들에 대하여 앞서 론의한 근사방법들은 정확한 결과를 주지 않을수 있다. 또한 다중적분은 심프손규칙이나 다른 근사방법들에 의하여 풀기가 어렵다. 이런 경우 몬떼까를로적분방법을 적용할수 있다. 용어 몬떼까를로는 결정론적인 문제들을 푸는데 우연수를 리용하는 방법에 적용된다.

그림 부-26에 보여 준것과 같은 함수의 적분은 몬떼까를로방법을 적용하여 a부터 b까지 구간의 $f(x)$를 포함하는 직4각형구역에서 n개의 우연위치들을 발생시켜 계산한다(그림 부-27). 그러면 적분에 대한 근사는

$$\int_a^b f(x)dx \approx h(b-a)\frac{n_{\text{count}}}{n} \qquad (부-91)$$

와 같이 계산된다. 여기서 파라메터 n_{count}는 $f(x)$와 x축사이에 있는 우연점들의 개수이다. 직4각형구역안의 우연위치 (x, y)는 우연수 r_1와 r_2을 발생시키고

$$h = y_{\max} - y_{\min}, \quad x = a + r_1(b-a), \quad y = y_{\min} + r_2 h \qquad (부-92)$$

을 수행하여 계산한다. 류사한 방법은 다중적분에도 적용할수 있다.

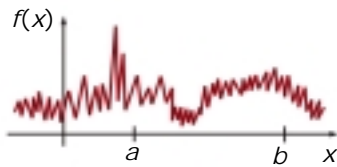



그림 부-26. 고주파진동을 가지는 함수

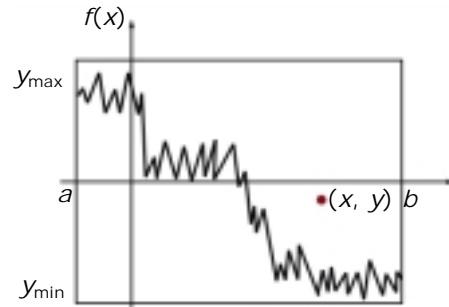



그림 부-27. 구간 (a, b)에서 함수 $f(x)$를 둘러싸는 직 4 각형구역

우연수 r_1와 r_2은 구간(0, 1)에서 균일분포한다. 우연수는 고수준언어의 우연수함수나 통계적프로그람으로부터 얻을수 있으며 또는 선형합동발생기(linear congruential generator)라고 부르는 다음의 알고리듬을 리용하여 얻을수 있다.

$$i_k = ai_{k-1} + c(\bmod m), \quad k = 1, 2, 3, \dots$$
$$r_k = \frac{i_k}{m} \qquad (부-93)$$

여기서 파라메터 a, c, m, i_0은 옹근수이며 i_0은 종자(seed)라고 하는 시작값이다. 파라메터 m은 주어진 콤퓨터에서 될수록 크게 선택하며 a와 c의 값들은 우연수렬이 길면서도 중복이 없도록 선택한다. 실례로 옹근수를 32bit로 표현하는 콤퓨터에서는 $m=2^{32}$, a=1664525, c=1013904223로 설정할수 있다.

자료모임에 대한 곡선맞추기

자료점들의 모임에 함수(선형 또는 비선형)를 맞추는 표준적인 방법은 최소두제곱알고리듬이다. 자료점들의 2차원모임 (x_k, y_k), k=1, 2, ⋯에 대하여 먼저 함수형식 $f(x)$를 선택하는데 그것은 선형함수, 다항식함수 또는 기타 곡선함수들이 될수 있다. 다음에 매 x_k에서 $f(x)$와 y_k값사이의 차를 결정하고 두제곱편차들의 합을 계산한다.

$$E = \sum_{k=1}^{n} \left[y_k - f(x_k) \right]^2 \qquad \text{(부-94)}$$

함수 $f(x)$의 파라메터들은 E가 최소이도록 결정한다. 실례로 선형함수

$$f(x) = a_0 + a_1 x$$

에 대하여 파라메터 a_0과 a_1에는 E를 최소화하는 값이 할당된다. a_0과 a_1의 값은 최소화요구의 결과인 련립2원1차방정식을 풀어 결정한다. 즉 E는 a_0에 대한 편도함수가 0이고 a_1에 대한 편도함수가 0이면 최소로 될것이다.

$$\frac{\partial E}{\partial a_0} = 0 , \qquad \frac{\partial E}{\partial a_1} = 0$$

류사한 계산은 다른 함수들에 대해서도 수행된다. 다항식

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ \cdots \ + a_n x^n$$

에 대하여 파라메터 a_k의 값들을 결정하는 련립n원1차방정식을 풀어야 한다. 최소두제곱맞추기는 매개 변수에 대하여 선형 또는 비선형일수 있는 다변수함수 $f(x_1, x_2, \ ... \ , x_m)$에도 적용할수 있다.

참고문헌

AKELEY, K. AND T. JERMOLUK (1988). "High-Performance Polygon Rendering", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 239-246.

AKELEY, K. (1993). "RealityEngine Graphics", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp.109-116.

AMANATIDES, J. (1984). "Ray Tracing with Cones", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp.129-135.

AMBURN, P., E. GRANT AND T. WHITTED (1986). "Managing Geometric Complexity with Enhanced Procedural Models", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 189-196.

ANJYO, K., Y. USAMI AND T. KURIHARA (1992). "A Simple Method for Extracting the Natural Beauty of Hair", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 111-120.

APPLE COMPUTER, INC. (1985). *Inside Macintosh,* Volume I, Addison-Wesley, Reading, MA.

APPLE COMPUTER, INC. (1987). *Human Interface Guidelines:*
The Apple Desktop Interface, Addison-Wesley, Reading, MA.

ARVO, J. AND D. KIRK (1987). "Fast Ray Tracing by Ray Classification", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 55-64.

ARVO, J. AND D. KIRK (1990). "Particle Transport and Image Synthesis", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 63-66.

ARVO, J., ED. (1991). *Graphics Gems II,* Academic Press, Inc., San Diego, CA.

ATHERTON, P. R. (1983). "A Scan-Line Hidden Surface Removal Procedure for Constructive Solid Geometry", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 73-82.

BARAFF, D. (1989). "Analytical Methods for Dynamic Simulation of Non-Penetrating Rigid Bodies", in proceedings of SIGGRAPH '89, *Computer Graphics, 23(3),* pp.
223-232.

BARAFF, D. AND A. WITKIN (1992). "Dynamic Simulation of Non-Penetrating Flexible Bodies", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 303-308.

BARKANS, A. C. (1990). "High-Speed, High-Quality, An-tialised Vector Generation", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 319-326.

BARNSLEY, M. F, A. JACQUIN, F. MALASSENT, ET AL. (1988). "Harnessing Chaos for Image Synthesis", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 131-140.

BARNSLEY, M. (1993). *Fractals Everywhere,* Second Edition, Academic Press, Inc., San Diego, CA.

BARR, A. H. (1981). "Superquadrics and Angle-Preserving Transformations", *IEEE Computer Graphics and Applications,* 1(1), pp. 11-23.

BARR, A. H. (1986). "Ray Tracing Deformed Surfaces", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 287-296.

BARSKY, B. A. AND J. C. BEATTY (1983). "Local Control of Bias and Tension in Beta-Splines", *ACM Transactions on Graphics,* 2(2), pp. 109-134.

BARSKY, B. A. (1984). "A Discription and Evaluation of Various 3-D Models", *IEEE Computer Graphics and Applications,* 4(1), pp. 38-52.

BARZEL, R. AND A. H. BARR (1988). "A Modeling System Based on Dynamic Constraints", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 179-188.

BARZEL, R. (1992). *Physically-Based Modeling for Computer Graphics,* Academic Press, Inc., San Diego, CA.

BAUM, D. R., S. MANN, K. P. SMITH, ET AL. (1991). "Making Radiosity Usable: Automatic Preprocessing and Meshing Techniques for the Generation of Accurate Radiosity Solutions", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 51-61.

BECKER, S. C., W. A. BARRETT, AND D. R. OLSEN JR. (1991). "Interactive Measurement of Three-Dimensional Objects Using a Depth Buffer and Linear Probe", ACM *Transactions on Graphics,* 10(2), pp. 201-207.

BECKER, B. G. AND N. L. MAX (1993). "Smooth Transitions between Bump-Rendering Algorithms", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 183-190.

BEIER, T. AND S. NEELY (1992). "Feature-Based Image Metamorphosis", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 35-42.

BERGMAN, L./ H. FUCHS, E. GRANT, ET AL. (1986). "Image Rendering by Adaptive Refinement", in proceedings of SIGGRAPH *'86, Computer Graphics,* 20(4), pp. 29-38.

BERGMAN, L. D., J. S. RICHARDSON, D. C. RICHARDSON, ET AL. (1993). "VIEW—an Epioratory Molecular Visualization System with User-Definable Interaction Sequences", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp.117-126.

BEZIER, P. (1972). *Numerical Control: Mathematics and Applications,* translated by A. R. Forrest and A. F. Pankhurst, John Wiley & Sons, London.

B IER, E. A., S. A. MACKAY, D. A. STEWART, ET AL. (1986).
"Snap-Dragging", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 241-248.

BIER, E. A., M. C. STONE, K. PIER, ET AL. (1993). "Toolglass and Magic Lenses: The See-Through Interface", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp.73-80.

BISHOP, G. AND D. M. WIEMER (1986). "Fast Phong Shading", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 103-106.

BLAKE, J. W. (1993). *PHIGS and PHIGS Plus,* Academic Press, London.

BLESER, T. (1988). "TAE Plus Styleguide User Interface Description", NASA Goddard Space Flight Center, Green-belt, MD.

BLINN, J. F. AND M. E. NEWELL (1976). "Texture and Reflection in Computer-Generated Images", CACM, 19(10), pp.542-547.

BLINN, J. F. (1977). "Models of Light Reflection for Computer-Synthesized Pictures", Computer Graphics, 11(2), pp.192-198.

BLINN, J. F. AND M. E. NEWELL (1978). "Clipping Using Homogeneous Coordinates", Computer Graphics, 12(3), pp.245-251.

BLINN, J. F. (1978). "Simulation of Wrinkled Surfaces", Computer Graphics, 12(3), pp. 286-292.

BLINN, J. F. (1982). "A Generalization of Algebraic Surface Drawing", *ACM Transactions on Graphics,* 1(3), pp. 235-256.

BLINN, J. F. (1982). "Light Reflection Functions for Simulation of Clouds and Dusty Surfaces", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 21-29.

BLINN, J. F. (1993). "A Trip Down the Graphics Pipeline:
The Homogeneous Perspective Transform", *IEEE Computer Graphics and Applications,* 13(3), pp. 75-80.

BLOOMENTHAL, J. (1985). "Modeling the Mighty Maple", in proceedings of SIGGRAPH '85, *Computer Graphics, 19(3),* pp. 305-312.

BONO, P. R., J. L. ENCARNACAO, F. R. A. HOPGOOD, ET AL. (1982). "GKS: The First Graphics Standard", *IEEE Computer Graphics and Applications,* 2(5), pp. 9-23.

BOOTH, K. S., M. P. BRYDEN, W. B. COWAN, ET AL. (1987). "On the Parameters of Human Visual Performance: An Investigation of the Benefits of Antialiasing", *IEEE Computer Graphics and Applications,* 7(9), pp. 34-41.

BRESENHAM, J. E. (1965). "Algorithm for Computer Control of A Digital Plotter", IBM Systems Journal, 4(1), pp. 25-30.

BRESENHAM, J. E. (1977). "A Linear Algorithm for Incremental Digital Display of Circular Arcs", CACM, 20(2), pp. 100-106.

BROOKS, F. P., JR. (1986). "Walkthrough: A Dynamic Graphics System for Simulating Virtual Buildings", Interactive 3D 1986.

BROOKS, F. P., JR. (1988). "Grasping Reality Through Illusion: Interactive Graphics Serving Science", CHI '88, pp. 1-11.

BROOKS, J., P. FREDERICK, M. OUH-YOUNG/ J. J. BATTER, ET AL. (1990). "Project GROPE - Haptic Display for Scientific Visualization", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), 24(4), pp. 177-185.

BROWN, M. H. AND R. SEDGEWICK (1984). "A System for Algorithm Animation", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 177-186.

BROWN, J. R. AND S. CUNNINGHAM (1989). *Programming the User Interface,* John Wiley & Sons, New York.

BRUDERLIN, A. AND T. W. CALVERT (1989). "Goal-Directed, Dynamic Animation of Human Walking", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 233-242.

BRUNET, P. AND I. NAVAZO (1990). "Solid Representation and Operation Using Extended Octrees", ACM *Transactions on Graphics,* 9(2), pp. 170-197.

BRYSON, S. AND C. LEVIT (1992). "The Virtual Wind Tunnel", *IEEE Computer Graphics and Applications,* 12(4), pp. 25-34.

BURT, P. J. AND E. H. ADELSON (1983). "A Multiresolution Spline with Application to Image Mosaics", ACM Transactions on Graphics, 2(4), pp. 217-236.

BUXTON, W., M. R. LAMB, D. SHERMAN, ET AL. (1983). "Towards a Comprehensive User Interface Management System", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 35-42.

BUXTON, W., R. HILL, AND P. ROWLEY (1985). "Issues and Techniques in Touch-Sensitive Tablet Input", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp.215-224.

CALVERT, T., A. BRUDERLIN/ J. DILL, ET AL. (1993). "Desktop Animation of Multiple Human Figures", *IEEE Computer Graphics and Applications,* 13(3), pp. 18-26.

CAMBELL, G., T. A. DEFANTI, J. FREDERIKSEN, ET AL. (1986). "Two Bit/Pixel Full-Color Encoding", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 215-224.

CAMPBELL, III., A. T. AND D. S. FUSSELL (1990). "Adaptive Mesh Generation for Global Diffuse Illumination", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 155-164.

CARD, S. K., J. D. MACKINLAY, AND G. G. ROBERTSON (1991). "The Information Visualizer, an Information Workspace", CHI '91, pp. 181-188.

CARIGNAN, M./ Y. YANG, N. M. THALMANN, ET AL. (1992). "Dressing Animated Synthetic Actors with Complex Deformable Clothes", in proceedings of SIGGRAPH *'92, Computer Graphics,* 26(2), pp. 99-104.

CARLBOM, I., I. CHAKRAVARTY, AND D. VANDERSCHEL (1985). *"A* Hierarchical Data Structure for Representing the Spatial Decomposition of 3-D Objects", *IEEE Computer Graphics and Applications,* 5(4), pp. 24-31.

CARPENTER, L. (1984). "The A-Buffer: An Antialiased Hidden-Surface Method", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 103-108.

CARROLL, J. M. AND C. CARRITHERS (1984). "Training Wheels in a User Interface", CACM, 27(8), pp. 800-806.

CASALE M. S. AND E. L. STANTON (1985). "An Overview of Analytic Solid Modeling", *IEEE Computer Graphics and Applications,* 5(2), pp. 45-56.

CATMULL, E. (1975). "Computer Display of Curved Surfaces", in proceedings of the IEEE Conference on Computer Graphics, *Pattern Recognition and Data Structures.* Also in Freeman (1980), PP. 309-315.

CATMULL/ E. (1984). "An Analytic Visible Surface Algorithm for Independent Pixel Processing", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 109-115.

CHAZELLE, B. AND J. INCERPI (1984). "Triangulation and Shape Complexity", *ACM Transactions on Graphics,* 3(2), pp.135-152.

CHEN, M., S. J. MOUNTFORD, AND A. SELLEN (1988). "A Study in Interactive 3D Rotation Using 2D Control Devices", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 121-130.

CHEN, S. E., H. E. RUSHMEIER, G. MILLER, ET AL. (1991). "A Progressive Multi-Pass Method for Global Illumination", in proceedings of SIGGRAPH '91, *Computer Grapliics,* 25(4), pp. 165-174.

CHIN, N. AND S. FEINER (1989). "Near Real-Time Shadow Generation Using BSP Trees", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 99-106.

CHUANG, R. AND G. ENTIS (1983). "3-D Shaded Computer Animation—Step by Step", *IEEE Computer Graphics and Applications,* 3(3), pp. 18-25.

CHUNG, J. C., ET AL. (1989). "Exploring Virtual Worlds with Head-Mounted Visual Displays", *Proceedings of SPIE Meeting on Non-Holographic True 3-Dimensional Displai/ Technologies,* 1083, January 1989, pp. 15-20.

CLARK, J. H. (1982). "The Geometry Engine: A VLSI Geometry System for Graphics", in proceedings of SIGGRAPH '82, *Computer Graphics, 16(3),* pp. 127-133.

COHEN, M. F. AND D. P. GREENBERG (1985). "The Hemi-Cube: A Radiosity Solution for Complex Environments", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 31-40.

COHEN, M. F, S. E. CHEN, J. R. WALLACE, ET AL. (1988). "A Progressive Refinement Approach to Fast Radiosity Image Generation", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 75-84.

COHEN, M. F. AND J. R. WALLACE (1993). *Radiosity and Realistic Image Synthesis,* Academic Press, Boston, MA. COOK, R. L. AND K. E. TORRANCE (1982). "A Reflectance
Model for Computer Graphics", ACM *Transactions on*

Graphics, 1(1), pp. 7-24.

COOK, R. L., T. PORTER, AND L. CARPENTER (1984). "Distributed Ray Tracing", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 137-145.

COOK, R. L. (1984). "Shade Trees", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 223-231.

COOK, R. L. (1986). "Stochastic Sampling in Computer Graphics", *ACM Transactions on Graphics,* 6(1), pp. 51-72.

COOK, R. L., L. CARPENTER, AND E. CATMULL (1987). "The Reyes Image Rendering Architecture", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 95-102.

COQUILLART, S. AND P. JANCENE (1991). "Animated Free-Form Deformation: An Interactive Animation Technique", in proceedings of SIGGRAPH '91, *Computer Grapliics,* 25(4), pp. 23-26.

CROW, F. C. (1977). "The Aliasing Problem in Computer-Synthesized Shaded Images", CACM, 20(11), pp. 799-805.

CROW, F. C. (1977). "Shadow Algorithms for Computer Graphics", in proceedings of SIGGRAPH '77, *Computer Graphics,* 11(2), pp. 242-248.

CROW, F. C. (1978). "The Use of Grayscale for Improved Raster Display of Vectors and Characters", in proceedings of SIGGRAPH '78, *Computer Graphics,* 12(3), pp. 1-5.

CROW, F. C. (1981). "A Comparison of Antialiasing Techniques", *IEEE Computer Graphics and Applications,* 1(1), pp.40-49.

CROW, F. C. (1982). "A More Flexible Image Generation Environment", in proceedings of SIGGRAPH '82, *Computer Grapliics,* 16(3), pp. 9-18.

CRUZ-NEIRA, C., D. J. SANDIN, AND T. A. DEFANTI (1993). "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE", in proceedings of SIGGRAPH '93, *Computer Grapliics Proceedings,* pp.135-142.

CUNNINGHAM, S., N. K. CRAICHILL, M. W. FONG, ET AL., ED.
(1992). *Computer Graphics Using Object-Oriented Programming,* John Wiley & Sons, New York.

CUTLER, E., D. GILLY, AND T. O'REILLY, ED. (1992). *The X Window System in a Nutshell,* Second Edition, O'Reilly & Assoc., Inc., Sebastopol, CA.

CYRUS, M. AND J. BECK (1978). "Generalized Two- and Three-Dimensional Clipping", Computers and Graphics, 3(1), pp. 23-28.

DAY, A. M. (1990). "The Implementation of an Algorithm to Find the Convex Hull of a Set of Three-Dimensional Points", *ACM Transactions on Graphics,* 9(1), pp. 105-132.

DE REFFYE, P., C. EDELIN, J. FRANCON, ET AL. (1988). "Plant Models Faithful to Botanical Structure and Development", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 151-158.

DEERING, M. (1992). "High Resolution Virtual Reality", in proceedings of SIGGRAPH *'92, Computer Graphics,* 26(2), pp. 195-202.

DEERING, M. F. AND S. R. NELSON (1993). "Leo: A System for Cost-Effective 3D Shaded Graphics", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 101-108.

DEMKO, S., L. HODGES, AND B. NAYLOR (1985). "Construction of Fractal Objects with Iterated Function Systems", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 271-278.

DEPP, S. W. AND W. E. HOWARD (1993). "Flat-Panel Displays", Scientific American, 266(3), pp. 90-97.

DEROSE, T. D. (1988). "Geometric Continuity, Shape Parameters, and Geometric Constructions for Catmull-Rom Splines", *ACM Transactions on Graphics,* 7(1), pp. 1-41.

DIGITAL EQUIPMENT CORP. (1989). "Digital Equipment Corporation XUI Style Guide", Maynard, MA.

DIPPE, M. AND J. SWENSEN (1984). "An Adaptive Subdivision Algorithm and Parallel Architecture for Realistic Image Synthesis", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 149-158.

DOBKIN, D., L. GUIBAS, J. HERSHBERGER, ET AL. (1988). "An Efficient Algorithm for Finding the CSG Representation of a Simple Polygon", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 31-40.

DOCTOR, L. J. AND J. G. TORBERG (1981). "Display Techniques for Octree-Encoded Objects", *IEEE Computer Graphics and Applications,* 1(3), pp. 29-38.

DORSEY, J. 0., F. X. SILLION, AND D. P. GREENBERG (1991).
"Design and Simulation of Opera Lighting and Projection Effects", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 41-50. DREBIN, R. A./ L. CARPENTER, AND P. HANRAHAN (1988). "Volume Rendering", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 65-74.

DUFF, T. (1985). "Compositing 3D Rendered Images", in
proceedings of SIGGRAPH '85, *Computer Graphics,*
19(3), pp. 41-44. DURRETT, H. J., ED. (1987). *Color and the Computer,* Academic
Press, Boston.

DUVANENKO, V. (1990). "Improved Line Segment Clipping", Dr. Dobb's Journal, July 1990.

DYER, S. AND S. WHITMAN (1987). "A Vectorized Scan-Line Z-Buffer Rendering Algorithm", *IEEE Computer Graphics and Applications,* 7(7), pp. 34-45.

DYER, S. (1990). "A Dataflow Toolkit for Visualization", *IEEE Computer Graphics and Applications,* 10(4), pp. 60-69.

EARNSHAW, R. A., ED. (1985). *Fundamental Algorithms for Computer Graphics,* Springer-Verlag, Berlin.

EDELSBRUNNER, H. (1987). *Algorithms in Computational Geometry,* Springer-Verlag, Berlin.

EDELSBRUNNER/ H. AND E. P. MUCKE (1990). "Simulation of Simplicity: A Technique to Cope with Degenerate Cases in Geometric Algorithms", *ACM Transactions on Graphics,* 9(1), pp. 66-104.

ELBER, G. AND E. COHEN (1990). "Hidden Curve Removal for Free Form Surfaces", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 95-104.

ENDERLE, G., K. KANSY, AND G. PFAFF (1984). *Computer Graphics Programming: GKS—The Graphics Standard,* Springer-Verlag, Berlin.

FARIN, G. (1988). *Curves and Surfaces for Computer Aided Geometric Design,* Academic Press, Boston, MA.

FAROUKI, R. T. AND J. K. HINDS (1985). "A Hierarchy of Geometric Forms", *IEEE Computer Graplncs and Applications,* 5(5), pp. 51-78.

FEDER, J. (1988). *Fractals,* Plenum Press, New York.

FEINER, S., S. NAGY, AND A. VAN DAM (1982). "An Experimental System for Creating and Presenting Interactive Graphical Documents", *ACM Transactions on Graphics,* 1(1), pp. 59-77.

FERWERDA, J. A. AND D. P. GREENBERG (1988). "A Psy-chophysical Approach to Assessing the Quality of An-tialiased Images", *IEEE Computer Graphics and Applications,* 8(5), pp. 85-95.

FISHKIN, K. P. AND B. A. BARSKY (1984). "A Family of New Algorithms for Soft Filling", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 235-244.

FIUME, E. L. (1989). *The Mathematical Structure of Raster Graphics,*

Academic Press, Boston.

FOLEY, J. D., V. L. WALLACE, AND P. CHAN (1984). "The Human Factors of Computer Graphics Interaction Techniques", *IEEE Computer Graphics and Applications,* 4(11), pp.13-48.

FOLEY, J. D. (1987). "Interfaces for Advanced Computing", Scientific American, 257(4), pp. 126-135.

FOLEY, J. D., A. VAN DAM, S. K. FEINER, ET AL. (1990). *Computer Grapliics: Principles and Practice,* Addison-Wesley, Reading, MA.

FOURNIER, A., D. FUSSEL, AND L. CARPENTER (1982). "Computer Rendering of Stochastic Models", CACM, 25(6), pp.371-384.

FOURNIER, A. AND D. Y. MONTUNO (1984). "Triangulating Simple Polygons and Equivalent Problems", *ACM Transactions on Grapliics,* 3(2), pp. 153-174.

FOURNIER, A. AND W. T. REEVES (1986). "A Simple Model of Ocean Waves", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 75-84.

FOURNIER, A. AND D. FUSSELL (1988). "On the Power of the Frame Buffer", *ACM Transactions on Graplncs,* 7(2), pp. 103-128.

FOURNIER, A. AND E. FIUME (1988). "Constant-Time Filtering with Space-Variant Kernels", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 229-238.

FOWLER, D. R., H. MEINHARDT, AND P. PRUSINKIEWICZ (1992). "Modeling Seashells", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 379-387.

FOX, D. AND M. WAITE (1984). *Computer Animation Primer,* McGraw-Hill, New York.

FRANCIS, G. K. (1987). A *Topological Picturebook,* Springer-Verlag, New York.

FRANKLIN, W. R. AND M. S. KANKANHALLI (1990). "Parallel Object-Space Hidden Surface Removal", in proceedings of SIGGRAPH *'90, Computer Graphics,* 24(4), pp. 87-94.

FREEMAN, H. ED. (1980). *Tutorial and Selected readings in Interactive Computer Graphics,* IEEE Computer Society Press, Silver Springs, MD.

FRENKEL, K. A. (1989). "Volume Rendering", CACM, 32(4), pp.426-435.

FRIEDER, G., D. GORDON, AND R. A. REYNOLD (1985). "Back-to-Front Display of Voxel-Based Objects", *IEEE Computer Graphics and Applications,* 5(1), pp. 52-60.

FRIEDHOFF, R. M. AND W. BENZON (1989). *The Second Computer Revolution: Visualization,* Harry N. Abrams, Inc., New York.

FUCHS, H., S. M. PIZER, E. R. HEINZ, S. H. BLOOMBER, L. TSAI, AND D. C. STRICKLAND (1982). "Design of and Image Editing with a Space-Filling Three-Dimensional Display Based on a Standard Raster Graphics System", *Proceedings of SPIE,* 367, August 1982, pp. 117-127.

FUCHS, H., J. POULTON, J. EYLES, ET AL. (1989). "Pixel-Planes 5: A Heterogeneous Multiprocessor Graphics System Using Processor-Enhanced Memories", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 79-88.

FUJIMOTO, A. AND K. IWATA (1983). "Jag-Free Images on Raster Displays", *IEEE Computer Graphics and Applications,* 3(9), pp. 26-34.

FUNKHOUSER, T. A. AND C. H. SEQUIN (1993). "Adaptive Display Algorithms for Interactive Frame Rates During Visualization Complex Virtual Environments", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 247-254.

GALYEAN, T. A. AND J. F. HUGHES (1991). "Sculpting: An Interactive Volumetric Modeling Technique", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 267-274.

GARDNER, G. Y. (1985). "Visual Simulation of Clouds", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 297-304.

GASCUEL, M.-P. (1993). "An Implicit Formulation for Precise Contact Modeling between Flexible Solids", in proceedings of SIGGRAPH '93, *Computer Graphics,* pp. 313-320.

GASKINS, T. (1992). *PHIGS Programming Manual,* O'Reilly & Associates, Sebastopol, CA.

GHARACHORLOO, N./ S. GUPTA, R. F. SPROULL, ET AL. (1989). "A Characterization of Ten Rasterization Algorithms", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 355-368.

GIRARD, M. (1987). "Interactive Design of 3D Computer-Animated Legged Animal Motion", *IEEE Computer Graphics and Applications,* 7(6), pp. 39-51.

GLASSNER, A. S. (1984). "Space Subdivision for Fast Ray Tracing", *IEEE Computer Graphics and Applications,* 4(10), pp.15-22.

GLASSNER, A. S. (1986). "Adaptive Precision in Texture Mapping", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 297-306.

GLASSNER, A. S. (1988). "Spacetime Ray Tracing for Animation", *IEEE Computer Graphics and Applications,* 8(2), pp. 60-70.

GLASSNER, A. S., ED. (1989). *An Introduction to Ray Tracing,* Academic Press, San Diego, CA.

GLASSNER, A. S./ ED. (1990). *Graphics Gems,* Academic Press, San Diego, CA.

GLASSNER, A. S. (1992). "Geometric Substitution: A Tutorial", *IEEE Computer Graphics and Applications,* 12(1), pp. 22-36.

GLASSNER, A. S. (1994). *Principles of Digital Image Synthesis,* Morgan-Kaufmann, Inc., New York.

GLEICHER, M. AND A. WITKIN (1992). "Through-the-Lens Camera Control", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 331-340.

GOLDSMITH, J. AND J. SALMON (1987). "Automatic Creation of Object Hierarchies for Ray Tracing", *IEEE Computer Graphics and Applications,* 7(5), pp. 14-20.

GONZALEZ, R. C. AND P. WINTZ (1987). *Digital Image Processing,* Addison-Wesley, Reading, MA.

GOOD, D. M., J. A. WHITESIDE, D. R. WIXON, AND S. J. JONES (1984). "Building A User-Derived Interface", CACM, 27(10), pp. 1032-1042.

GOODMAN, T. AND R. SPENCE (1978). "The Effect of System Response Time on Interactive Computer-Aided Problem Solving", in proceedings of SIGGRAPH '78, *Computer Graphics,* 12(3), pp. 100-104.

GORAL, C. M., K. E. TORRANCE, D. P. GREENBERG, ET AL.
(1984). "Modeling the Interaction of Light Between Diffuse Surfaces", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 213-222.

GORDON, D. AND S. CHEN (1991). "Front-to-Back Display of BSP Trees", *IEEE Computer Graphics and Applications,* 11(5), pp. 79-85.

GORTLER, S. J., P. SCHRODER, M. F. COHEN, ET AL. (1993).
"Wavelet Radiosity", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 221-230.

GREEN, M. (1985). "The University of Alberta User Interface Management System", in proceedings of SIGGRAPH
'85, *Computer Graphics,* 19(3), pp. 205-214.

GREENE, N., M. KASS, AND G. MILLER (1993). "Hierarchical Z-Buffer Visibility", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 231-238.

HAEBERLI, P. AND K. AKELEY (1990). "The Accumulation Buffer: Hardware Support for High-Quality Rendering", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 309-318.

HAHN, J. K. (1988). "Realistic Animation of Rigid Bodies", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 299-308.

HALL, R. A. AND D. P. GREENBERG (1983). "A Testbed for Realistic Image Synthesis", *IEEE Computer Graphics and Applications,* 3(8), pp. 10-20.

HALL, R. (1989). *Illumination and Color in Computer Generated Imagery,* Springer-Verlag, New York.

HANRAHAN, P. (1982). "Creating Volume Models from Edge-Vertex Graphs", in proceedings of SIGGRAPH *'82, Computer Graphics,* 16(3), pp. 77-84.

HANRAHAN, P. AND J. LAWSON (1990). "A Language for Shading and Lighting Calculations", in proceedings of SIGGRAPH *'90, Computer Graphics,* 24(4), pp. 289-298.

HART, J. C, D. J. SANDIN, AND L. H. KAUFFMAN (1989). "Ray Tracing Deterministic 3D Fractals", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 289-296.

HART, J. C. AND T. A. DEFANTI (1991). "Efficient Antialiased Rendering of 3—D Linear Fractals", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 91-100.

HE, X. D., P. 0. HEYNEN, R. L. PHILLIPS, ET AL. (1992). "A Fast and Accurate Light Reflection Model", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 253-254.

HEARN, D. AND P. BAKER (1991). "Scientific Visualization:
An Introduction", *Eurographics '91 Technical Report Series,* Tutorial Lecture 6.

HECKBERT, P. (1982). "Color Image Quantization for Frame Buffer Display", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 297-307.

HECKBERT/ P. AND P. HANRAHAN (1984). "Beam Tracing Polygonal Objects", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 119-127.

HOPGOOD, F. R. A., D. A. DUCE, J. R. GALLOP, ET AL. (1983). *Introduction to the Graphical Kernel System* (GXS), Academic Press, London.

HOPGOOD/ F. R. A. AND D. A. DUCE (1991). A *Primer for PHIGS,* John Wiley & Sons, Chichester, England.

HOPPE, H., T. DEROSE, T. MCDONALD/ ET AL. (1993). "Mesh Optimization", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 19-26.

HOWARD, T. L. *].,* W. T. HEWITT, R. J. HUBBOLD, ET AL. (1991). A *Practical Introduction to PHIGS and PHIGS Plus,* Addi-son-Wesley, Wokingham, England.

HUGHES, J. F. (1992). "Scheduled Fourier Volume Morph-ing", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 43-46.

HUITRIC, H. AND M. NAHAS (1985). "B-Spline Surfaces: A Tool for Computer Painting", *IEEE Computer Graphics and Applications,* 5(3), pp. 39-47.

IKEDO, T. (1984). "High-Speed Techniques for a 3-D Color Graphics Terminal", *IEEE Computer Graphics and Applications,* 4(5), pp. 46-58.

IMMEL, D. S., M. F. COHEN, AND D. P. GREENBERG (1986). "A Radiosity Method for Non-Diffuse Environments", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 133-142.

ISAACS, P. M. AND M. F. COHEN (1987). "Controlling Dynamic Simulation with Kinematic Constraints, Behavior Functions, and Inverse Dynamics", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 215-224.

JARVIS, J. F, C. N. JUDICE, AND W. H. NINKE (1976). "A Survey of Techniques for the Image Display of Continuous Tone Pictures on Bilevel Displays", *Computer Graphics and Image Processing,* 5(1), pp. 13-40.

JOHNSON, S. A. (1982). "Clinical Varifocal Mirror Display System at the University of Utah", *Proceedings of SPIE,* 367, August 1982, pp. 145-148.

KAJIYA, J. T. (1983). "New Techniques for Ray Tracing Pro-cedurally Defined Objects", ACM *Transactions on Graphics, 2(3), pp. 161-181.*

KAJIYA, J. T. (1986). "The Rendering Equation", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 143-150.

KAJIYA/ J. T. AND T. L. KAY (1989). "Rendering Fur with Three-Dimensional Textures", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 271-280.

KAPPEL, M. R. (1985). "An Ellipse-Drawing Algorithm for Faster Displays", in *Fundamental Algorithms for Computer Graphics,* Springer-Verlag, Berlin, pp. 257-280.

KARASICK, M., D. LIEBER, AND L. R. NACKMAN (1991). "Efficient Delaunay Triangulation Using Rational Arithmetic", ACM *Transactions on Graphics,* 10(1), pp. 71-91.

KASS, M. (1992). "CONDOR: Constraint-Based Dataflow", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 321-330.

KASSON, J. M. AND W. PLOUFFE (1992). "An Analysis of Selected Computer Interchange Color Spaces", ACM *Transactions on Graphics,* 11(4), pp. 373-405.

KAUFMAN, A. (1987). "Efficient Algorithms for 3D Scan-Conversion of Parametric Curves, Surfaces, and Volumes", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 171-179.

KAWAGUCHI, Y. (1982). "A Morphological Study of the Form of Nature", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 223-232.

KAY, T. L. AND J. T. KAJIYA (1986). "Ray Tracing Complex Scenes", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 269-278.

KAY, D. C. AND J. R. LEVINE (1992). *Graphics File Formats,* Windcrest/McGraw-Hill, New York.

KELLEY, A. D., M. C. MALIN, AND G. M. NIELSON (1988). "Terrain Simulation using a Model of Stream Erosion", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 263-268.

KENT, J. R., W. E. CARLSON, AND R. E. PARENT (1992). "Shape Transformation for Polyhedral Objects", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 47-54.

KIRK, D. AND J. ARVO (1991). "Unbiased Sampling Techniques for Image Synthesis", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 153-156.

KIRK, D., ED. (1992). *Graphics Gems III,* Academic Press, San Diego, CA.

KNUTH, D. E. (1987). "Digital Halftones by Dot Diffusion", ACM *Transactions on Graphics,* 6(4), pp. 245-273.

KOCHANEK, D. H. U. AND R. H. BARTELS (1984). "Interpolating Splines with Local Tension, Continuity, and Bias Control", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 33-41.

KOH, E.-K. AND D. HEARN (1992). "Fast Generation and Surface Structuring Methods for Terrain and Other Natural Phenomena", in proceedings of Eurographs *'92 Computer Graphics Forum,* 11(3), pp. C-169-180.

KORIEN, J. U. AND N. I. BADLER (1982). "Techniques for Generating the Goal-Directed Motion of Articulated Structures", *IEEE Computer Graphics and Applications, 2(9),* pp. 71-81.

KORIEN, J. U. AND N. I. BADLER (1983). "Temporal antialiasing in Computer-Generated Animation", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 377-388.

LASSETER, J. (1987). "Principles of Traditional Animation Applied to 3D Computer Animation", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 35-44.

LAUR, D. AND P. HANRAHAN (1991). "Hierarchical Splatting:
A Progressive Refinement Algorithm for Volume Rendering", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 285-288.

LAUREL, B. (1990). *The Art of Human-Computer Interface Design,* Addision-Wesley, Reading, MA.

LEE, M. E., R. A. REDNER, AND S. P. USELTON (1985). "Statisi-
cally Optimized Sampling for Distributed Ray Tracing",
in proceedings of SIGGRAPH '85, *Computer Graphics,*
19(3), pp. 61-68. LEVINTHAL, A. AND T. PORTER (1984). "CHAP — A SIMD
Graphics Processor", in proceedings of SIGGRAPH '84,
Computer Graphics, 18(3), pp. 77-82.

LEVOY, M. (1988). "Display of Surfaces from Volume Data", *IEEE Computer Graphics and Applications,* 8(3), pp. 29-37.

LEVOY, M. (1990). "A Hybrid Ray Tracer for Rendering Polygon and Volume Data", *IEEE Computer Graphics and Applications,* 10(2), pp. 33-40.

LEWIS, J.-P. (1989). "Algorithms for Solid Noise Synthesis", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 263-270.

LIANG, Y.-D. AND B. A. BARSKY (1983). "An Analysis and Algorithm for Polygon Clipping." CACM, 26(11), pp. 868-877.

LIANG, Y.-D. AND B. A. BARSKY (1984). "A New Concept and Method for Line Clipping", *ACM Transactions on Graphics,* 3(1), pp. 1-22.

LIEN, S.-L., M. SHANTZ, AND V. PRATT (1987). "Adaptive Forward Differencing for Rendering Curves and Surfaces",
in proceedings of SIGGRAPH '87, *Computer Graphics,*
21(4), pp. 111-118. LINDLEY, C. A. (1992). *Practical Ray Tracing in C,* John Wiley
& Sons, New York. LISCHINSKI, D., F. TAMPIERI, AND D. P. GREENBERG (1993).
"Combining Hierarchical Radiosity and Discontinuity
Meshing", in proceedings of SIGGRAPH '93, *Computer*
Graphics, pp. 199-208. LITWINOWICZ, P. C. (1991). "Inkwell: A 2 1/2-D Animation
System", in proceedings of SIGGRAPH '91, *Computer*
Graphics, 25(4), pp. 113-122.

LODDING, K. N. (1983). "Iconic Interfacing", *IEEE Computer Graphics and Applications,* 3(2), pp. 11-20.

LOKE, T.-S., D. TAN, H.-S. SEAH, ET AL. (1992). "Rendering Fireworks Displays", *IEEE Computer Graphics and Applications,* 12(3), pp. 33-43.

LOOMIS, J., H. POIZNER, U. BELLUGI, ET AL. (1983). "Computer Graphic Modeling of American Sign Language", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp.105-114.

LORENSON, W. E. AND H. CLINE (1987). "Marching Cubes: A High-Resolution 3D Surface Construction Algorithm", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21 (4), pp. 163-169.

MACKINLAY, J. D., S. K. CARD, AND G. G. ROBERTSON (1990). "Rapid Controlled Movement Through a Virtual 3D Workspace", SIGGRAPH 90, pp. 171-176.

MACKINLAY, J. D., G. G. ROBERTSON, AND S. K. CARD (1991). "The Perspective Wall: Detail and Context Smoothly Integrated", CHI '91, pp. 173-179.

MAGNENAT-THALMANN, N. AND D. THALMANN (1985). *Computer Animation: Theory and Practice,* Springer-Verlag, Tokyo.

MAGNENAT-THALMANN, N. AND D. THALMANN (1987). *Image Synthesis,* Springer-Verlag, Tokyo.

MAGNENAT-THALMANN, N. AND D. THALMANN (1991). "Complex Models for Animating Synthetic Actors", *IEEE Computer Graphics and Applications,* 11(5), pp.
32-45.

MANDELBROT, B. B. (1977). *Fractals: Form, Chance, and Dimension,* Freeman Press, San Francisco.

MANDELBROT, B. B. (1982). *The Fractal Geometry of Nature,* Freeman Press, New York.

MANTYLA, M. (1988). *An Introduction to Solid Modeling,* Computer Science Press, Rockville, MD.

MAX, N. L. AND D. M. LERNER (1985). "A Two-and-a-Half-D Motion Blur Algorithm", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 85-94.

MAX, N. L. (1986). "Atmospheric Illumination and Shadows", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 117-124.

MAX, N. L. (1990). "Cone-Spheres", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 59-62.

METAXAS, D. AND D. TERZOPOULOS (1992). "Dynamic Deformation of Solid Primitives with Constraints", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp.309-312.

MEYER, G. W., H. E. RUSHMEIER, M. F. COHEN, ET AL. (1986). "An Experimental Evaluation of Computer Graphics Imagery", ACM *Transactions on Graphics,* 6(1), pp. 30-50.

MEYER, G. W. AND D. P. GREENBERG (1988). "Color-Defective Vision and Computer Graphics Displays", *IEEE Computer Graphics and Applications, 8(5),* pp. 28-40.

MEYERS, D., S. SKINNER, AND K. SLOAN (1992). "Surfaces from Contours", *ACM Transactions on Graphics,* 11(3), pp.228-258.

MILLER, G. S. P. (1988). "The Motion Dynamics of Snakes and Worms", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 169-178.

MILLER, J. V., D. E. BREEN, W. E. LORENSON, ET AL. (1991). "Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 217-226.

MITCHELL, D. P. (1991). "Spectrally Optimal Sampling for Distribution Ray Tracing", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 157-165.

MITCHELL, D. P. AND P. HANRAHAN (1992). "Illumination from Curved Reflectors", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 283-291.

MIYATA, K. (1990). "A Method of Generating Stone Wall Patterns", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 387-394.

MOLNAR, S./ J. EYLES, AND J. POULTON (1992). "PixelFlow:
High-Speed Rendering Using Image Composition", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 231-240.

MOON, F. C. (1992). *Chaotic and Fractal Dynamics,* John Wiley & Sons, New York.

MOORE, M. AND J. WILHELMS (1988). "Collision Detection and Response for Computer Animation", in proceedings of SIGGRAPH

'88, *Computer Graphics,* 22(4), pp. 289-298.

MORTENSON, M. E. (1985). *Geometric Modeling,* John Wiley & Sons, New York.

MURAKI, S. (1991). "Volumetric Shape Description of Range Data Using the 'Blobby Model' ", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 227-235.

MUSGRAVE, F. K., C. E. KOLB, AND R. S. MACE (1989). "The Synthesis and Rendering of Eroded Fractal Terrains", in proceedings of SIGGRAPH '89, *Computer Graphics*, 23(3), pp. 41-50.

MYERS, B. A. AND W. BUXTON (1986). "Creating High-Interactive and Graphical User Interfaces by Demonstration", in proceedings of SIGGRAPH '86, *Computer Graphics*, 20(4), pp. 249-258.

NAYLOR, B., J. AMANATIDES, AND W. THIBAULT (1990). "Merging BSP Trees Yields Polyhedral Set Operations", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp.115-124.

NEWMAN, W. H. (1968). "A System for Interactive Graphical Programming", *SJCC, Thompson Books,* Washington, D. C., pp. 47-54.

NEWMAN, W. H. AND R. F. SPROULL (1979). *Principles of Interactive Computer Graphics,* McGraw-Hill, New York.

NGO, J. T. AND J. MARKS (1993). "Spacetime Constraints Revisited", in proceedings of SIGGRAPH '93, *Computer Graphics,* pp. 343-350.

NICHOLL, T. M., D. T. LEE, AND R. A. NICHOLL (1987). "An Efficient New Algorithm for 2D Line Clipping: Its Development and Analysis", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 253-262.

NIELSON, G. M., B. SHRIVER, AND L. ROSENBLUM, ED. (1990).
Visualization in Scientific Computing, IEEE Computer Society Press, Los Alamitos, CA.

NIELSON. G. M. (1993). "Scattered Data Modeling", *IEEE Computer Graphics and Applications,* 13(1), pp. 60-70.

NISHIMURA, H. (1985). "Object Modeling by Distribution Function and a Method of Image Generation", Journal Electronics Comm. Conf. '85, J68(4), pp. 718-725.

NISHITA, T. AND E. NAKAMAE (1986). "Continuous-Tone Representation of Three-Dimensional Objects Illuminated by Sky Light", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 125-132.

NISHITA, T, T. SIRAI, K. TADAMURA, ET AL. (1993). "Display of the Earth Taking into Account Atmospheric Scattering", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 175-182.

NORTON, A. (1982). "Generation and Display of Geometric Fractals in 3-D", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 61-67.

NSFINVITATIONAL WORKSHOP (1992). "Research Directions in Virtual Environments", Computer Graphics, 26(3), pp.153-177.

OKABE, H., H. IMAOKA, T. TOMIHA, ET AL. (1992). "Three-Dimensional Apparel CAD System", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 105-110.

OPENGL ARCHITECTURE REVIEW BOARD (1993). *OpenGL Programming Guide,* Addision-Wesley, Reading, MA.

OPPENHEIMER, P. E. (1986). "Real-Time Design and Animation of Fractal Plants and Trees", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 55-64.

OSF/MOTIF (1989). *OSF/Motif Style Guide,* Open Software Foundation, Prentice-Hall, Englewood Cliffs, NJ.

PAINTER, J. AND K. SLOAN (1989). "Antialiased Ray Tracing by Adaptive Progressive Refinement", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 281-288.

PANG, A. T. (1990). "Line-Drawing Algorithms for Parallel Machines", *IEEE Computer Graphics and Applications,* 10(5), pp. 54-59.

PAVLIDIS, T. (1982). *Algorithms For Graphics and Image Processing,* Computer Science Press, Rockville, MD.

PAVLIDIS, T. (1983). "Curve Fitting with Conic Splines", *ACM Transactions on Graphics,* 2(1), pp. 1-31.

PEACHEY, D. R. (1986). "Modeling Waves and Surf", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 65-74.

PEITGEN, H.-O. AND P. H. RICHTER (1986). *The Beauty of Fractals,* Springer-Verlag, Berlin.

PEITGEN, H.-O. AND D. SAUPE, ED. (1988). *The Science of Fractal Images,* Springer-Verlag, Berlin.

PENTLAND, A. AND J. WILLIAMS (1989). "Good Vibrations:
Modal Dynamics for Graphics and Animation", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp.215-222.

PERLIN, K. AND E. M. HOFFERT (1989). "Hypertexture", in proceedings of SIGGRAPH '89, *Computer Graphics*, 23(3), pp. 253-262.

PHILLIPS, R. L. (1977). "A Query Language for a Network Data Base with Graphical Entities", in proceedings of SIGGRAPH '77, *Computer Graphics,* 11(2), pp. 179-185.

PHONG, B. T. (1975). "Illumination for Computer-Generated Images", CACM, 18(6), pp. 311-317.

PINEDA, J. (1988). "A Parallel Algorithm for Polygon Rasterization", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 17-20.

PTTEWAY, M. L. V. AND D. J. WATKINSON (1980). "Bresen-ham's Algorithm with Gray Scale", CACM, 23(11), pp. 625-626.

PLATT, J. C. AND A. H. BARR (1988). "Constraint Methods for Flexible Models", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 279-288.

PORTER, T. AND T. DUFF (1984). "Compositing Digital Images", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 253-259.

POTMESIL, M. AND I. CHAKRAVARTY (1982). "Synthetic Image Generation with a Lens and Aperture Camera Model", *ACM Transactions on Graphics, 1(2),* pp. 85-108.

POTMESIL, M. AND I. CHAKRAVARTY (1983). "Modeling Motion Blur in Computer-Generated Images", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 389-399.

POTMESIL, M. AND E. M. HOFFERT (1987). "FRAMES: Software Tools for Modeling, Rendering and Animation of 3D Scenes", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 85-93.

POTMESIL, M. AND E. M. HOFFERT (1989). "The Pixel Machine: A Parallel Image Computer", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 69-78.

PRATT, W. K. (1). *Digital Image Processing,* John Wiley & Sons, New York.

PREPARATA, F. P. AND M. I. SHAMOS (1985). *Computational Geometry,* Springer-Verlag, New York.

PRESS, W. H., S. A. TEUKOLSKY, W. T. VETTERLING, ET AL. (1992). *Numerical Recipes in C,* Cambridge University Press, Cambridge, England.

PRUSINKIEWICZ, P., M. S. HAMMEL, AND E. MJOLSNESS (1993). "Animation of Plant Development", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings*, pp. 351-360.

PRUYN, P. W. AND D. P. GREENBERG (1993). "Exploring 3D Computer Graphics in Cockpit Avionics", *IEEE Computer Graphics and Applications,* 13(3), pp. 28-35.

QUEK, L.-H. AND D. HEARN (1988). "Efficient Space-Subdivision

Methods in Ray-Tracing Algorithms", University of Illinois, Department of Computer Science Report UIUCDCS-R-88-1468.

RAIBERT, M. H. AND}. K. HODGINS (1991). "Animation of Dynamic Legged Locomotion", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 349-358.

REEVES, W. T. (1983). "Particle Systems: A Technique for Modeling a Class of Fuzzy Objects", *ACM Transactions on Graphics,* 2(2), pp. 91-108.

REEVES, W. T. (1983). "Particle Systems—A Technique for Modeling a Class of Fuzzy Objects", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 359-376.

REEVES, W. T. AND R. BLAU (1985). "Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 313-321.

REEVES, W. T., D. H. SALESIN, AND R. L. COOK (1987). "Rendering Antialiased Shadows with Depth Maps", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 283-291.

REQUICHA, A. A. G. AND J. R. ROSSIGNAC (1992). "Solid Modeling and Beyond", *IEEE Computer Graphics and Applications,* 12(5), pp. 31-44.

REYNOLDS, C. W. (1982). "Computer Animation with Scripts and Actors", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 289-296.

REYNOLDS, C. W. (1987). "Flocks, Herds, and Schools: A Distributed Behavioral Model", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 25-34.

RIESENFELD, R. F. (1981). "Homogeneous Coordinates and Projective Planes in Computer Graphics", *IEEE Computer Graphics and Applications,* 1(1), pp. 50-55.

ROBERTSON, P. K. (1988). "Visualizing Color Gamuts: A User Interface for the Effective Use of Perceptual Color Spaces in Data Displays", *IEEE Computer Graphics and Applications,* 8(5), pp. 50-64.

ROBERTSON, G. G., J. D. MACKINLAY AND S. K. CARD (1991). "Cone Trees: Animated 3D Visualizations of Hierarchical Information", CHI '91, pp. 189-194.

ROGERS, D. F. AND R. A. EARNSHAW, ED. (1987). *Techniques for Computer Graphics,* Springer-Verlag, New York.

ROGERS, D. F. ANDJ. A. ADAMS (1990). *Mathematical Elements for Computer Graphics,* McGraw-Hill, New York.

ROSENTHAL, D. S. H., ET AL. (1982). "The Detailed Semantics of Graphics Input Devices", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 33-38.

RUBINE, D. (1991). "Specifying Gestures by Example", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 329-337.

RUSHMEIER, H. AND K. TORRANCE (1987). "The Zonal Method for Calculating Light Intensities in the Presence of a Participating Medium", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 293-302.

RUSHMEIER, H. E. AND K. E. TORRANCE (1990). "Extending the Radiosity Method to Include Specularly Reflecting and Translucent Materials", *ACM Transactions on Graphics,* 9(1), pp. 1-27.

SABELLA, P. (1988). "A Rendering Algorithm for Visualizing 3D Scalar Fields", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 51-58.

SABIN, M. A. (1985). "Contouring: The State of the Art", in *Fundamental Algorithms for Computer Graphics,* R. A. Earnshaw, ed. Springer-Verlag, Berlin, pp. 411-482.

SALESIN, D. AND R. BARZEL (1993). "Adjustable Tools: An Object-Oriented Interaction Metaphor", ACM *Transactions on Graphics,* 12(1), pp. 103-107.

SAMET, H. AND R. E. WEBBER (1985). "Sorting a Collection of Polygons using Quadtrees", ACM *Transactions on Graphics,* 4(3), pp. 182-222.

SAMET, H. AND M. TAMMINEN (1985). "Bintrees, CSG Trees, and Time", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 121-130.

SAMET, H. AND R. E. WEBBER (1988). "Hierarchical Data Structures and Algorithms for Computer Graphics: Part I", *IEEE Computer Graphics and Applications,* 8(4), pp. 59-75.

SAMET. H. AND R. E. WEBBER (1988). "Hierarchical Data Structures and Algorithms for Computer Graphics: Part 2", *IEEE Computer Graphics and Applications,* 8(3), pp. 48-68.

SCHEIFLER, R. W. AND J. GETTYS (1986). "The X Window System", ACM *Transactions on Graphics,* 5(2), pp. 79-109.

SCHOENEMAN, C., J. DORSEY, B. SMITS, ET AL. (1993). "Global Illumination", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 143-146.

SCHRODER, P. AND P. HANRAHAN (1993). "On the Form Factor Between Two Polygons", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 163-164.

SCHWARTZ, M. W., W. B. COWAN, AND J. C. BEATTY (1987). "An Experimental Comparison of RGB, YIQ, LAB, HSV, and Opponent Color Models", ACM *Transactions on Graphics,* 6(2), pp. 123-158.

SEDERBERG, T. W. AND E. GREENWOOD (1992). "A Physically Based Approached to 2-D Shape Bending", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 25-34.

SEDERBERG, T. W, P. GAO, G. WANG, ET AL. (1993). "2D Shape Blending: An Intrinsic Solution to the Vertex Path Problem", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 15-18.

SEGAL, M. (1990). "Using Tolerances to Guarantee Valid Polyhedral Modeling Results", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 105-114.

SEGAL, M., C. KOROBKIN, R. VAN WIDENFELT, ET AL. (1992).
"Fast Shadows and Lighting Effects Using Texture Mapping", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 249-252.

SEQUIN, C. H. AND E. K. SMYRL (1989). "Parameterized Ray-Tracing", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 307-314.

SHERR, S. (1993). *Electronic Displays,* John Wiley & Sons, New York.

SHILLING, A. AND W. STRASSER (1993). "EXACT: Algorithm and Hardware Architecture for an Improved A-Buffer", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 85-92.

SHIRLEY, P. (1990). "A Ray Tracing Method for Illumination Calculation in Diffuse-Specular Scenes", Graphics Interface '90, pp. 205-212.

SHNEIDERMAN, B. (1986). *Designing the User Interface,* Addi-son-Wesley, Reading, MA.

SHOEMAKE, K. (1985). "Animating Rotation with Quaternion Curves", in proceedings of SIGGRAPH '85, *Computer Graphics,* 19(3), pp. 245-254.

SIBERT, J. L., W. D. HURLEY, AND T. W. BLESER (1986). "An Ob-ject-Oriented User Interface Management System", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 259-268.

SILLION, F. X. AND C. PUECH (1989). "A General Two-Pass Method Integrating Specular and Diffuse Reflection", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 335-344.

SILLION, F. X., J. R. ARVO, S. H. WESTIN, ET AL. (1991). "A Global

Illumination Solution for General Reflectance Distributions", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 187-196.

SIMS, K. (1990). "Particle Animation and Rendering Using Data Parallel Computation", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 405-413.

SIMS, K. (1991). "Artificial Evolution for Computer Graphics", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 319-328.

SINGH, B., J. C. BEATTY, K. S. BOOTH, ET AL. (1983). "A Graphics Editor for Benesh Movement Notation", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 51-62.

SMITH, A. R. (1978). "Color Gamut Transform Pairs", Computer Graphics, 12(3), pp. 12-19.

SMITH, A. R. (1979). "Tint Fill", Computer Graphics, 13(2), pp.276-283.

SMITH, A. R. (1984). "Plants, Fractals, and Formal Languages", in proceedings of SIGGRAPH '84, *Computer Graphics,* 18(3), pp. 1-10.

SMITH, R. B. (1987). "Experiences with the Alternate Reality Kit: An Example of the Tension Between Literalism and Magic", *IEEE Computer Graphics and Applications,* 7(9), pp.42-50.

SMITH, A. R. (1987). "Planar 2-Pass Texture Mapping and Warping", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 263-272.

SMITS, B. E., J. R. ARVO, AND D. H. SALESIN (1992). "An Importance-Driven Radiosity Algorithm", in proceedings of SIGGRAPH '92, *Computer Graphics, 2.6(2),* pp. 273-282.

SNYDER, J. M. AND J. T. KAJIYA (1992). "Generative Modeling: A Symbolic System for Geometric Modeling", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 369-378.

SNYDER, J. M., A. R. WOODBURY, K. FLEISCHER, ET AL. (1993). "Interval Method for Multi-Point Collisions between Time-Dependent Curved Surfaces", in proceedings of SIGGRAPH '93, *Computer Graphics,* pp. 321-334.

SPROULL, R. F. AND I. E. SUTHERLAND (1968). "A Clipping Divider", AFIPS Fall Joint Computer Conference.

STAM, J. AND E. FIUME (1993). "Turbulent Wind Fields for Gaseous Phenomena", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 369-376.

STETTNER, A. AND D. P. GREENBERG (1989). "Computer Graphics Visualization for Acoustic Simulation", in proceedings of SIGGRAPH '89, *Computer Graphics,* 23(3), pp.195-206.

STRASSMANN, S. (1986). "Hairy Brushes", in proceedings of SIGGRAPH *'86, Computer Graphics,* 20(4), pp. 225-232.

STRAUSS, P. S. AND R. CAREY (1992). "An Object-Oriented 3D Graphics Toolkit", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 341-349.

SUNG, H. C. K., G. ROGERS, AND W. J. KUBITZ (1990). "A Critical Evaluation of PEX", *IEEE Computer Graphics and Applications,* 10(6), pp. 65-75.

SUTHERLAND, I. E. (1963). "Sketchpad: A Man-Machine Graphical Communication System", AFIPS Spring Joint Computer Conference, 23 pp. 329-346.

SUTHERLAND, I. E., R. F. SPROULL, AND R. SCHUMACKER (1974). "A Characterization of Ten Hidden Surface Algorithms", ACM Computing Surveys, 6(1), pp. 1-55.

SUTHERLAND, I. E. AND G. W. HODGMAN (1974). "Reentrant Polygon Clipping", CACM, 17(1), pp. 32-42.

SWEZEY, R. W. AND E. G. DAVIS (1983). "A Case Study of Human Factors Guidelines in Computer Graphics", *IEEE Computer Graphics and Applications,* 3(8), pp. 21-30.

TAKALA, T. AND J. HAHN (1992). "Sound Rendering", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp.211-220.

TANNAS, *].,* LAWRENCE E., ED. (1985). *Flat-Panel Displays and CRTs,* Van Nostrand Reinhold Company, New York.

TELLER, S. AND P. HANRAHAN (1993). "Global Visibility Algorithms for Illumination Computations", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp.239-246.

TERZOPOULOS, D., J. PLATT, A. H. BARR, ET AL. (1987). "Elasti-cally Deformable Models", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 205-214.

THALMANN, D., ED. (1990). *Scientific Visualization and Graphics Simulation,* John Wiley & Sons, Chichester, England.

THIBAULT, W. C. AND B. F. NAYLOR (1987). "Set Operations on Polyhedra using Binary Space Partitioning Trees", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 153-162.

TORBERG, J. G. (1987). "A Parallel Processor Architecture for Graphics Arithmetic Operations", in proceedings of SIGGRAPH '87, *Computer Graphics,* 21(4), pp. 197-204.

TORRANCE, K. E. AND E. M. SPARROW (1967). "Theory for Off-Specular Reflection from Roughened Surfaces", *Optical Society of America,* 57(9), pp. 1105-1114.

TRAVIS, D. (1991). *Effective Color Displays,* Academic Press, London.

TUFTE, E. R. (1983). *The Visual Display of Quantitative Information,* Graphics Press, Cheshire, CN.

TUFTE, E. R. (1990). *Envisioning Information,* Graphics Press, Cheshire, CN.

TURKOWSKI, K. (1982). "Antialiasing Through the Use of Coordinate Transformations", ACM Transactions on Graphics, 1(3), pp. 215-234.

UPSON, C. AND M. KEELER (1988). "VBUFFER: Visible Volume Rendering", in proceedings of SIGGRAPH '88, *Computer Graphics,* 22(4), pp. 59-64.

UPSON, C., T. FAULHABER JR., D. KAMINS, ET AL. (1989). "The Application Visualization System: A Computational Environment for Scientific Visualization", *IEEE Computer Graphics and Applications,* 9(4), pp. 30-42.

UPSTILL, S. (1990). *The RenderMan Companion,* Addison-Wesley, Reading, MA.

VAN DE PANNE, M. AND E. FIUME (1993). "Sensor-Actuator Networks", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 335-342.

VAN WIJK, J. J. (1991). "Spot Noise-Texture Synthesis for Data Visualization", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 309-318.

VEENSTRA, J. AND N. AHUJA (1988). "Line Drawings of Octree-Represented Objects", ACM *Transactions on Graphics,* 7(1), pp. 61-75.

VELHO, L. AND J. D. M. GOMES (1991). "Digital Halftoning with Space-Filling Curves", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 81-90.

VON HERZEN, B., A. H. BARR, AND H. R. ZATZ (1990). "Geometric Collisions for Time-Dependent Parametric Surfaces", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 39-48.

WALLACE, V. L. (1976). "The Semantics of Graphic Input Devices", in proceedings of SIGGRAPH '76, *Computer Graphics,* 10(1), pp. 61-65.

WALLACE, J. R., K. A. ELMQUIST, AND E. A. HAINES (1989). "A Ray-Tracing Algorithm for Progressive Radiosity", in proceedings

of SIGGRAPH '89, *Computer Graphics,* 23(3), pp. 315-324.

WANGER, L. R., J. A. FERWERDA, AND D. P. GREENBERG (1992). "Perceiving Spatial Relationships in Computer-Generated Images", *IEEE Computer Graphics and Applications,* 12(3), pp. 44-58.

WARE, C. (1988). "Color Sequences for Univariate Maps:
Theory, Experiments, and Principles", *IEEE Computer Graphics and Applications,* 8(5), pp. 41-49.

WARN, D. R. (1983). "Lighting Controls for Synthetic Images", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 13-21.

WARNOCK, J. AND D. K. WYATT (1982). "A Device-Independent Graphics Imaging Model for Use with Raster Devices", in proceedings of SIGGRAPH '82, *Computer Graphics,* 16(3), pp. 313-319.

WATT, A. (1989). *Fundamentals of Three-Dimensional Computer Graphics,* Addison-Wesley, Wokingham, England.

WATT, M. (1990). "Light-Water Interaction Using Backward Beam Tracing", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 377-386.

WATT, A. AND M. WATT (1992). *Advanced Animation and Rendering Techniques,* Addison-Wesley/ Wokingham, England.

WEGHORST, H., G. HOOPER, AND D. P. GREENBERG (1984). "Improved Computational Methods for Ray Tracing", *ACM Transactions on Graphics,* 3(1), pp. 52-69.

WEIL, J. (1986). "The Synthesis of Cloth Objects", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp.49-54.

WEILER, K. AND P. ATHERTON (1977). "Hidden-Surface Removal Using Polygon Area Sorting", in proceedings of SIGGRAPH 77, *Computer Graphics,* 11(2), pp. 214-222.

WEILER, K. (1980). "Polygon Comparison Using a Graph Representation", in proceedings of SIGGRAPH '80, *Computer Graphics,* 14(3), pp. 10-18.

WESTIN, S. H., J. R. ARVO, AND K. E. TORRANCE (1992). "Predicting Reflectance Functions from Complex Surfaces", in proceedings of SIGGRAPH '92, *Computer Graphics,* 26(2), pp. 255-264.

WESTOVER, L. (1990). "Footprint Evaluation for Volume Rendering", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 367-376.

WHITTED, T. (1980). "An Improved Illumination Model for Shaded Display", CACM, 23(6), pp. 343-349.

WHITTED, T. AND D. M. WEIMER (1982). "A Software Testbed for the Development of 3D Raster Graphics Systems", ACM *Transactions on Graphics,* 1(1), pp. 43-58.

WHITTED, T. (1983). "Antialiased Line Drawing Using Brush Extrusion", in proceedings of SIGGRAPH '83, *Computer Graphics,* 17(3), pp. 151-156.

WILHELMS, J. (1987). "Toward Automatic Motion Control", *IEEE Computer Graphics and Applications,* 7(4), pp. 11-22.

WILHELMS, J. AND A. V. GELDER (1991). "A Coherent Projection Approach for Direct Volume Rendering", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp.275-284.

WILHELMS. J. AND A. VAN GELDER (1992). "Octrees for Faster Isosurface Generation", ACM *Transactions on Graphics,* 11(3), pp. 201-227.

WILLIAMS, L. (1990). "Performance-Driven Facial Animation", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 235-242.

WILLIAMS, P. L. (1992). "Visibility Ordering Meshed Polyhe-dra", ACM *Transactions on Graphics,* 11(2), pp. 103-126.

WITKIN, A. AND W. WELCH (1990). "Fast Animation and Control of Nonrigid Structures", in proceedings of SIGGRAPH '90, *Computer Graphics,* 24(4), pp. 243-252.

WITKIN, A. AND M. KASS (1991). "Reaction-Diffusion Textures", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 299-308.

WOLFRAM, S. (1991). *Mathematica,* Addison-Wesley, Reading, MA.

Woo, A., P. POULIN, AND A. FOURNIER (1990). "A Survey of Shadow Algorithms", *IEEE Computer Graphics and Applications,* 10(6), pp. 13-32.

WRIGHT, W. E. (1990). "Parallelization of Bresenham's Line and Circle Algorithms", *IEEE Computer Graphics and Applications,* 10(5), pp. 60-67.

Wu, X. (1991). "An Efficient Antialiasing Technique", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 143-152.

WYSZECKI, G. AND W. S. STILES (1982). *Color Science,* John Wiley & Sons, New York.

WYVILL, G., B. WYVILL, AND C. MCPHEETERS (1987). "Solid
Texturing of Soft Objects", *IEEE Computer Graphics and Applications,* 7(12), pp. 20-26.

YAEGER, L., C. UPSON, AND R. MYERS (1986). "Combining Physical and Visual Simulation: Creation of the Planet Jupiter for the Film "2010"", in proceedings of SIGGRAPH '86, *Computer Graphics,* 20(4), pp. 85-94.

YAGEL, R., D. COHEN, AND A. KAUFMAN (1992). "Discrete Ray Tracing", *IEEE Computer Graphics and Applications,* 12(5), pp. 19-28.

YAMAGUCHI, K./ T. L. KUNII, AND FUJIMURA (1984). "Octree-Related Data Structures and Algorithms", *IEEE Computer Graphics and Applications,* 4(1), pp. 53-59.

YOUNG, D. A. (1990). *The X Window System - Programming and Applications ivith Xt, OSF/Motif Edition,* Prentice-Hall, Englewood Cliffs, NJ.

ZELEZNICK, R. C., D. B. CONNER, M. M. WLOKA, ET AL. (1991). "An Object-Oriented Framework for the Integration of Interactive Animation Techniques", in proceedings of SIGGRAPH '91, *Computer Graphics,* 25(4), pp. 105-112.

ZELTZER, D. (1982). "Motor Control Techniques for Figure Animation", *IEEE Computer Graphics and Applications,* 2(9), pp. 53-60.

ZHANG, Y. AND R. E. WEBBER (1993). "Space Diffusion: An Improved Parallel Halftoning Technique Using Space-Filling Curves", in proceedings of SIGGRAPH '93, *Computer Graphics Proceedings,* pp. 305-312.

주제색인

ㄱ

ㄴ

ㄷ

ㄹ

ㅂ

ㅈ

ㅊ

ㅎ

ㄲ

ㅃ

ㅆ

ㅉ

ㅇ

＊＊＊

＊＊＊

함수색인

T

U